Patrick GUEULLE

Montages à composants programmables

EPROM, E2PROM, PAL, CARTES A PUCE, AUTOMATISMES, SYNTHESE DE SON, CONTROLE D'ACCES, JEUX...

Applications pratiques

ETSF EDITIONS TECHNIQUES ET SCIENTIFIQUES FRANÇAISES

Patrick GUEULLE

Ingénieur EFREI

Montages à COMPOSANTS PROGRAMMABLES

ETSF

EDITIONS TECHNIQUES ET SCIENTIFIQUES FRANÇAISES

2-12, rue de Bellevue - 75940 Paris Cedex 19
Tél : (1) 44 84 84 84

DANGER
LE PHOTOCOPILLAGE TUE LE LIVRE

Ce pictogramme mérite une explication. Son objet est d'alerter le lecteur sur la menace que représente pour l'avenir de l'écrit, particulièrement dans le domaine de l'édition technique et universitaire, le développement massif du ***photocopillage***.

Le Code de la propriété intellectuelle du 1er juillet 1992 interdit en effet expressément la photocopie à usage collectif sans autorisation des ayants droit. Or, cette pratique s'est généralisée dans les établissements d'enseignement supérieur, provoquant une baisse brutale des achats de livres et de revues, au point que la possibilité même pour les auteurs de créer des œuvres nouvelles et de les faire éditer correctement est aujourd'hui menacée.

Nous rappelons donc que toute reproduction, partielle ou totale, de la présente publication est interdite sans autorisation du Centre français d'exploitation du droit de copie (CFC, 3 rue Hautefeuille, 75006 Paris).

© Photographie couverture : Eric MALEMANCHE
© Photographies intérieur : Patrick GUEULLE

© E.T.S.F., Paris, 1993

ISBN 2-85535-210-X

Toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause est illicite selon le Code de la propriété intellectuelle (Art L 122-4) et constitue une contrefaçon réprimée par le Code pénal. • Seules sont autorisées (Art L 122-5) les copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective, ainsi que les analyses et courtes citations justifiées par le caractère critique, pédagogique ou d'information de l'œuvre à laquelle elles sont incorporées, sous réserve, toutefois, du respect des dispositions des articles L 122-10 à L 122-12 du même Code, relative à la reproduction par reprographie.

AVANT-PROPOS

Dans notre livre ***Composants électroniques programmables***, nous avons fait découvrir à nos lecteurs les principaux composants dont on peut modifier durablement le contenu grâce à des programmateurs faciles à construire: mémoires EPROM, PAL, microcontrôleurs, et même cartes à puce.

Avec ce nouvel ouvrage, voici venu le temps des applications pratiques de ces composants personnalisables dont vous avez appris à faire de véritables circuits intégrés spécifiques.

Bien entendu, nous avons sélectionné des montages qui seraient difficiles à développer à partir de composants ordinaires, ou dont le schéma se trouve spectaculairement simplifié par l'emploi de circuits intégrés programmables.

Nos domaines de prédilection vont donc être ici l'automatisation de toutes sortes de cycles, la synthèse de sons, ou la sécurité: contrôle d'accès, protection de logiciels, etc.

Mais nous nous offrirons tout de même quelques petites "récréations" comme la construction d'un jeu de dés électronique, ou la lecture du crédit des télécartes entamées!

TABLE DES MATIERES

Avant-propos 5

① Les composants programmables dans les montages pratiques 9

② Contrôleurs programmables 12

Un séquenceur à 8 sorties 12

Un mini-automate programmable 19

Un "microcontrôleur " à Z80 26

③ Synthétiseurs de son personnalisables 39

Un digitaliseur de son pour PC et compatibles 41

Un reproducteur de contrôle pour PC 49

Un reproducteur de son autonome à EPROM 54

Un système de développement pour synthèse de sons 59

Un reproducteur de son à UM 5100 73

Un synthétiseur vocal à UM 5100 82

④ Montages à cartes à puce 91

Un connecteur pour carte à puce 92

Un mini-lecteur de télécartes 95

Les logiciels d'exploitation 99

Une serrure électronique à télécarte 103

Un adaptateur pour clef à puce 108

Une serrure électronique à EEPROM 111

Une carte à puce à EEPROM 111

Une clef à EEPROM 118

Les logiciels d'exploitation 119

Un dongle à puce de télécarte 128

⑤ Applications des PAL 137

Un double dé électronique 137

Un décodeur 7 segments personnalisé 146

① LES COMPOSANTS PROGRAMMABLES DANS LES MONTAGES PRATIQUES

Utiliser des composants programmables revient plus ou moins à fabriquer soi-même des circuits intégrés qui n'existent pas dans le commerce, cela à l'aide d'un matériel qui peut souvent être simple et peu coûteux (typiquement un petit accessoire raccordé à un micro-ordinateur compatible PC).

Les composants programmables *vierges* sont facilement disponibles dans le commerce, prêts à subir de profondes modifications internes : chargement de données s'il s'agit de mémoires non volatiles, ou même câblage d'un schéma dans le cas des *réseaux logiques programmables*.

Ces modifications sont en principe définitives, et survivent notamment aux mises hors tension des composants : on transforme donc bel et bien des composants "génériques" en circuits intégrés spécifiques qui peuvent alors être manipulés tout à fait normalement.

Mais bien des composants programmables sont *effaçables* : soit par exposition à un rayonnement ultraviolet, soit électriquement, soit tout simplement par une nouvelle programmation qui "écrase" la précédente. On peut donc sans inconvénient mettre au point des montages par essais successifs, leur apporter des améliorations de temps à autre, ou modifier leur programmation plus ou moins souvent en fonction des nécessités de l'application.

Enfin, la récupération de composants programmables dans des réalisations devenues inutiles est monnaie courante : un simple effacement et ils sont à nouveau pratiquement comme neufs !

A partir du moment où on dispose des programmateurs appropriés, il est souvent plus économique, plus simple, et plus rapide de personnaliser des composants programmables que de réunir des circuits intégrés standards par un câblage qui peut parfois être touffu.

Cela, du moins, pour certaines catégories de réalisations et plus spécialement pour celles faisant appel à des circuits digitaux.

Mais dans beaucoup de cas, la solution optimale consiste finalement à associer composants programmables et circuits intégrés standards.

Les composants programmables sont particulièrement à l'aise dans les montages dont certaines fonctionnalités doivent pouvoir être rapidement modifiées : plutôt que de reprendre tout ou partie du câblage, il est infiniment plus confortable de remplacer une mémoire ou un PAL par une pièce équivalente, mais programmée différemment.

Dans les ***contrôleurs programmables***, tout d'abord, qui permettent de commander un certain nombre d'organes selon un cycle bien précis, en tenant au besoin compte d'évènements détectés par des capteurs appropriés.

Dans les ***synthétiseurs de son***, également, qui sont capables de reproduire indéfiniment et sans aucune usure, des sons absolument quelconques préalablement "enregistrés" dans des mémoires non volatiles.

Quant aux ***systèmes de sécurité*** en tous genres, la programmation d'un code secret approprié dans une carte à puce ou dans une mémoire débrochable permet de "montrer patte blanche" de façon simple et sûre.

Grâce aux composants programmables, il devient même possible de développer des montages plus ou moins "passe-partout", que l'on peut réaliser en plusieurs exemplaires et adapter en un temps record à une application bien spécifique par simple personnalisation logicielle d'un ou quelques composants.

Bien évidemment, ces perspectives n'ont pas échappé aux fabricants de "kits" : il devient fréquent que des composants programmables (et bien sûr programmés !) soient "maquillés" en circuits intégrés ordinaires par simple effacement ou camouflage de leur marquage.

On fait alors volontiers appel à des composants programmables "OTP" (*One Time Programmable*) qui, présentés en boîtier plastique, sont bien plus économiques que les versions effaçables à boîtier céramique.

Un avantage supplémentaire réside dans le fait que certains composants programmables peuvent être "verrouillés" en fin de programmation : il devient alors impossible de recopier leur contenu, et donc de contrefaire le montage...

Il n'est pas rare non plus que des revues d'électronique publient des réalisations faisant appel à des composants programmables qui sont alors vendus, déjà personnalisés, par un circuit commercial mis en place à cet effet.

Le lecteur peut alors accéder à la technologie des composants programmables sans avoir à s'équiper d'un programmateur, ce qui demeure par contre nécessaire lorsque le contenu du composant est fourni sous la forme d'un listing ou d'une disquette.

Les montages décrits dans cet ouvrage, pour leur part, seront accompagnés de toutes les données permettant de mener à bien leur réalisation au moyen des programmateurs décrits dans notre livre "***composants électroniques programmables***", déjà paru dans la même collection.

② CONTRÔLEURS PROGRAMMABLES

Un contrôleur programmable est un montage électronique plus ou moins complexe, qui assure le *pilotage automatique* d'un appareil ou d'une installation selon des règles préalablement définies et enregistrées dans sa mémoire.

Ces règles peuvent se réduire à un simple cycle qui devra se dérouler de façon immuable (le contrôleur sera alors un *séquenceur*), ou comporter des variantes selon les évènements externes susceptibles de se produire (on sera alors en présence d'un *automate programmable* ou d'un *microcontrôleur*).

Dans les deux cas, il est particulièrement indiqué d'utiliser des composants programmables pour enregistrer les instructions à exécuter.

UN SÉQUENCEUR À 8 SORTIES

Bien des problèmes d'automatisation peuvent se ramener à la commande d'un certain nombre d'organes *tout ou rien* selon un cycle bien défini.

Ce genre de tâche est à la portée d'un simple programmateur électromécanique à cames (comme on en trouve sur les machines à laver), mais on peut faire beaucoup mieux par des moyens purement électroniques !

La mémoire EPROM est le composant programmable idéal pour ce genre d'application : une 2764, par exemple, peut abriter 8192 groupes de huit bits pouvant être programmés individuellement à 1 ou à 0.

Chacun de ces groupes de bits peut être lu, sur huit broches dites *de données*, en appliquant son "numéro" exprimé en binaire, sur un certain nombre de broches dites *d'adresse.*

La 2764, pour sa part, possède treize lignes d'adresse numérotées de A0 à A12 : un rapide calcul confirme que 2 à la puissance 13 est bien égal à 8192.

Il suffit de relier ces lignes d'adresse aux sorties d'un compteur binaire pour faire défiler le contenu de la mémoire sur les broches de données, au rythme de l'horloge pilotant le compteur. Dès lors, il n'y a plus qu'à programmer la mémoire en fonction du comportement que l'on désire obtenir au niveau de chaque sortie.

Selon le problème à résoudre, on peut faire défiler tout le contenu de la mémoire en une fraction de seconde ou en plusieurs heures, avec de larges possibilités de variation de vitesse parfaitement inaccessibles aux programmateurs à moteur synchrone.

Et s'il faut changer de *programme*, quoi de plus simple que de remplacer l'EPROM par une autre programmée en conséquence?

Notre schéma de la figure 2.1 a été conçu dans le but d'offrir un maximum de souplesse, et donc de larges possibilités de personnalisation de ses applications.

L'horloge chargée de faire défiler les adresses est équipée d'un circuit CMOS de type 4060, qui regroupe un diviseur de fréquence et un oscillateur à quartz ou RC. Associé à un quartz fort courant et même facile à récupérer sur une montre hors d'usage (32,768 kHz), celui.ci fournit simultanément dix cadences d'impulsions extrêmement précises allant d'une demi seconde à 0,48 millisecondes.

Avec un condensateur et deux résistances à la place du quartz, rien n'interdit d'atteindre une impulsion toutes les quelques heures s'il le faut, avec une précision moindre il est vrai.

Dans les deux cas, le simple déplacement d'une connexion permet de choisir instantanément entre dix vitesses de déroulement du cycle.

Les impulsions d'horloge attaquent un compteur CMOS de type 4040, qui possède 12 sorties. Nous avons affecté la dernière à l'arrêt du générateur d'horloge (par sa broche RESET), ce qui en laisse 11 disponibles pour la commande de l'EPROM.

Avec une 2764, deux lignes d'adresse restent donc libres, sur lesquelles nous avons prévu d'agir manuellement par deux cavaliers de sélection + 5 V / masse : quatre combinaisons sont possibles, qui correspondent à quatre *programmes* distincts enregistrés dans la même EPROM.

Chaque programme ne comptera évidemment que 2048 pas, mais c'est amplement suffisant pour la plupart des applications : à raison de deux pas par seconde, le cycle durera 17 minutes, tandis qu'il pourra durer 24 heures si l'horloge délivre une impulsion toutes les 42 secondes.

Le même montage pourra donc servir aussi bien au pilotage d'un réseau de chemin de fer miniature qu'à la simulation de présence dans une habitation, sans parler d'une foule d'autres usages qui ne dépendent que de votre imagination !

Le même schéma conviendrait également à des mémoires 2732 (deux programmes de 2048 pas) ou 2716 (un seul programme), mais l'intérêt de la manoeuvre n'est pas évident du fait que ces références tendent à coûter plus cher que la 2764 !

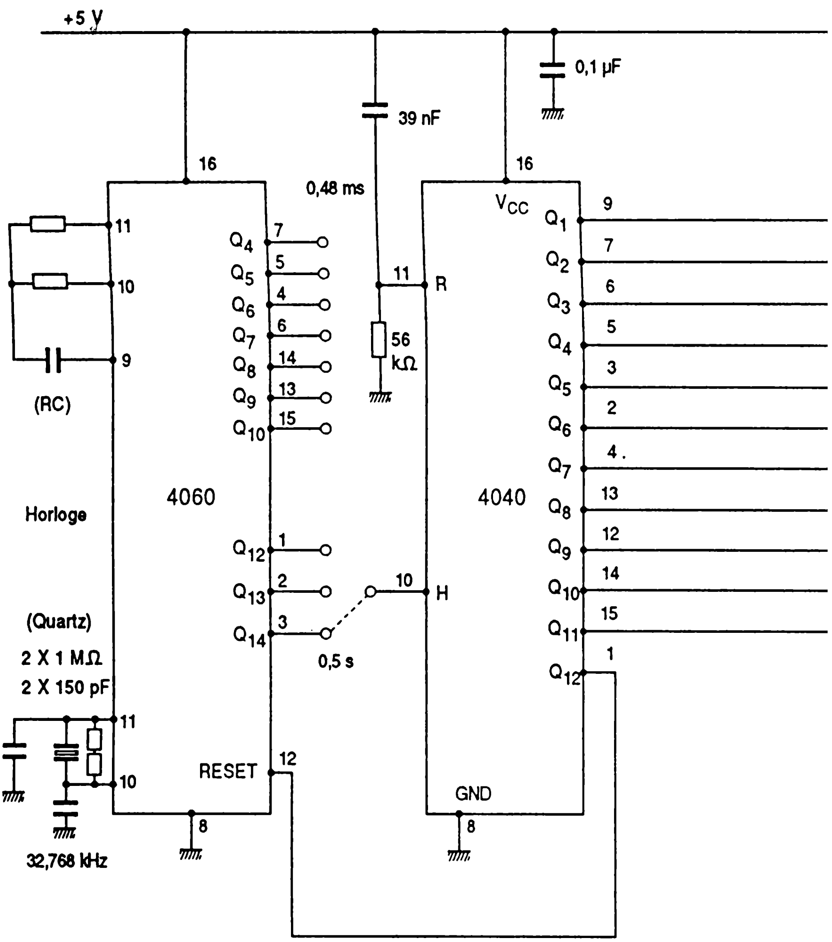
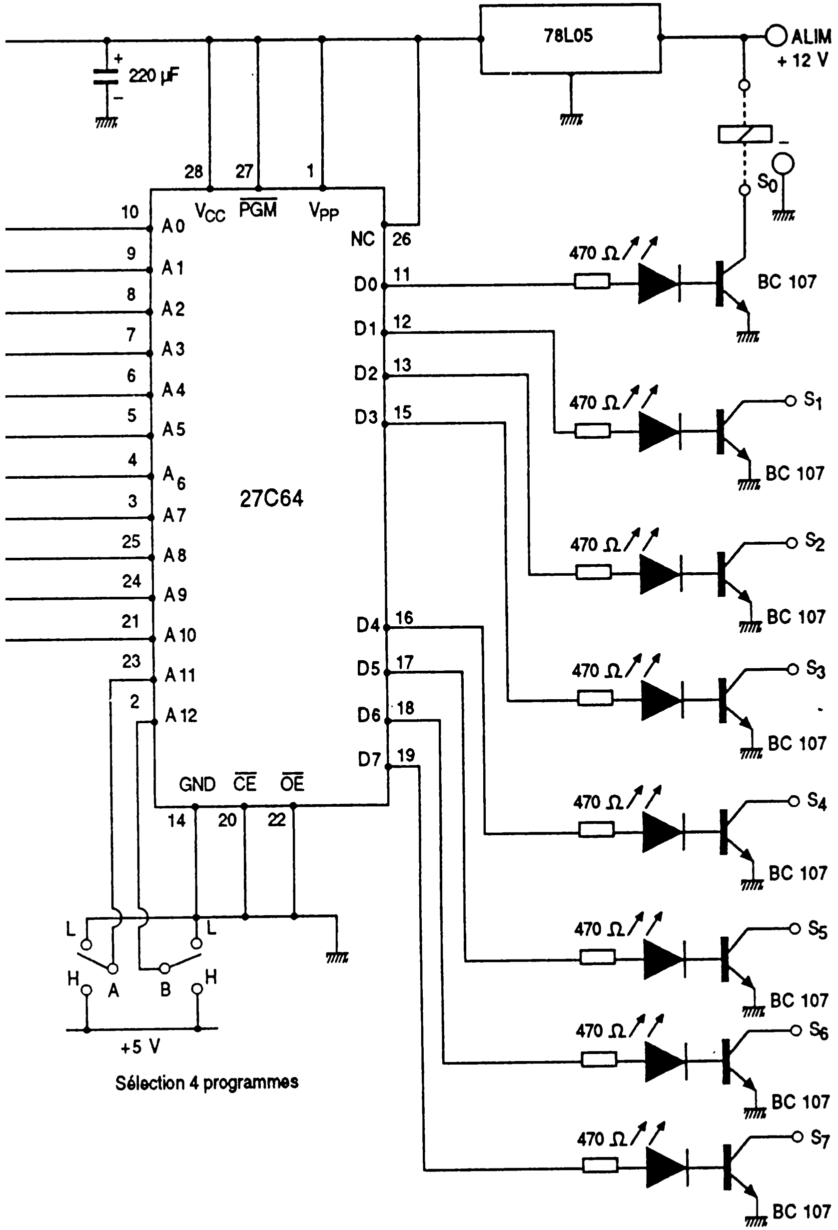


Fig. 2.1. – Schéma du séquenceur.


78L05
ALIM
+ 12 V
220 µF
28
27
1
V_{CC}
$\overline{PGM}$
V_{PP}
NC
26
S_0
470 Ω
BC 107
S_1
S_2
S_3
S_4
S_5
S_6
S_7
10 A 0
9 A 1
8 A 2
7 A 3
6 A 4
5 A 5
4 A_6
3 A 7
25 A 8
24 A 9
21 A 10
23 A 11
2 A 12
27C64
D0 11
D1 12
D2 13
D3 15
D4 16
D5 17
D6 18
D7 19
GND
$\overline{CE}$
$\overline{OE}$
14
20
22
L
H
A
B
+5 V
Sélection 4 programmes

Les huit sorties de données de la mémoire commandent des transistors montés en "collecteur ouvert", par l'intermédiaire de diodes LED servant de voyants de contrôle. On pourra leur faire alimenter des relais ou directement de petits moteurs à courant continu ou des solénoïdes, sous réserve de prévoir des composants d'absorption des surtensions de commutation (diodes ou condensateurs).

L'alimentation de la carte pourra se faire entre 6 et 24 V environ (typiquement 12 V) grâce à un régulateur 78L05 incorporé qui fournit du 5 V à la mémoire et à ses composants associés.

Un réseau 39 nF / 56 kΩ assure la remise à zéro du compteur 4040, afin que le cycle démarre toujours à son début lors de la mise sous tension du montage.

La personnalisation du cycle nécessite le remplissage de l'EPROM avec des informations binaires correspondant, pour chaque adresse, aux états désirés sur les sorties.

Les programmateurs d'EPROM purement manuels possèdent huit interrupteurs (un par bit de données) qu'il faut positionner selon les valeurs à enregistrer : sans commentaire !

Les appareils plus élaborés, notamment ceux adaptables sur micro-ordinateur, attendent pour chaque adresse une valeur numérique comprise entre 0 et 255 en décimal (0 et FF en hexadécimal). Cette valeur se calcule en additionnant les *poids* de toutes les sorties devant être actives (transistor conducteur et voyant allumé) en même temps.

Les poids respectifs des huit sorties sont les suivants : 1 pour S0, 2 pour S1, 4 pour S2, 8 pour S3, 16 pour S4, 32 pour S5, 64 pour S6, et 128 pour S7.

0 correspond donc bien à toutes les sorties inactives, et 255 à toutes les sorties actives !

Bien entendu, rien n'empêche de programmer plusieurs adresses consécutives avec le même contenu : on obtiendra alors un état dont la durée sera un multiple de la cadence de l'horloge. C'est d'ailleurs là le plus simple moyen à employer lorsque le nombre de pas de programme disponible se révèle excessif : en quadruplant chaque pas, quitte à accélérer l'horloge d'un facteur quatre, on limite le cycle à 512 pas. Mais on peut évidemment *démultiplier* encore plus...

La réalisation pratique du séquenceur fait appel à un circuit imprimé très compact, et donc facile à incorporer à toutes sortes de projets, dont le tracé est fourni à la figure 2.2.

Une fois gravé, on le percera à 1 mm et on câblera les composants selon le plan de la figure 2.3, sans oublier les cinq straps en fil rigide qui ont été nécessaires pour éviter le recours à une carte double face, ni le cavalier en fil souple servant à sélectionner la fréquence d'horloge parmi les dix possibles.

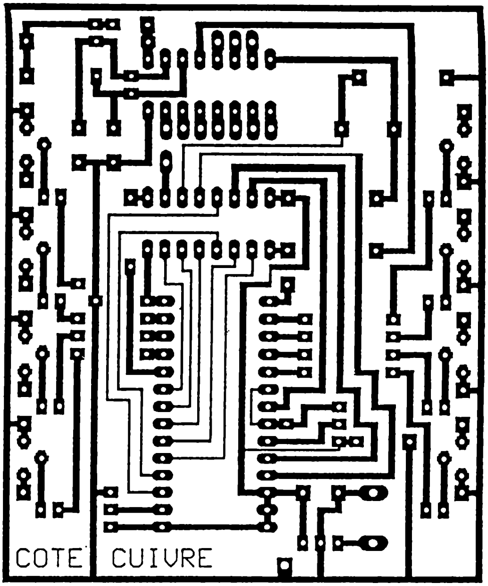


Fig. 2.2. – Circuit imprimé du séquenceur.

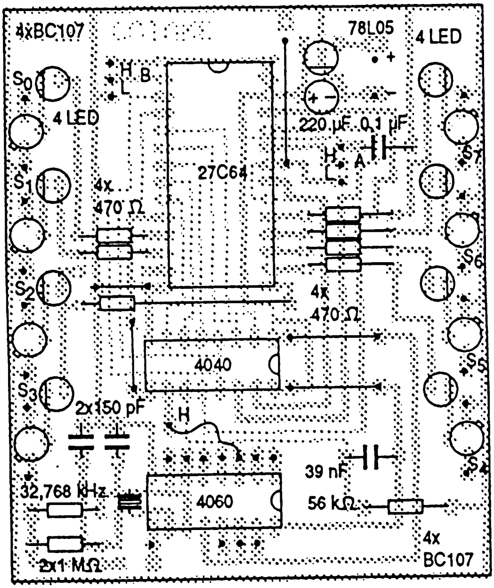


Fig. 2.3. – Plan de câblage du séquenceur.

Il est souhaitable de prévoir un support de bonne qualité (type "tulipe") pour l'EPROM, afin de faciliter son échange lors de la mise au point des programmes. Rappelons en effet que ce type de mémoire peut s'effacer aux ultraviolets et se reprogrammer des centaines de fois. Il serait dommage de s'en priver...

Liste des composants du séquenceur

Résistances (5% 1/4 W) :
R1 : 1 MΩ,
R2 : 1 MΩ,
R3 : 56 kΩ,
R4 à R 11 : 470 Ω.

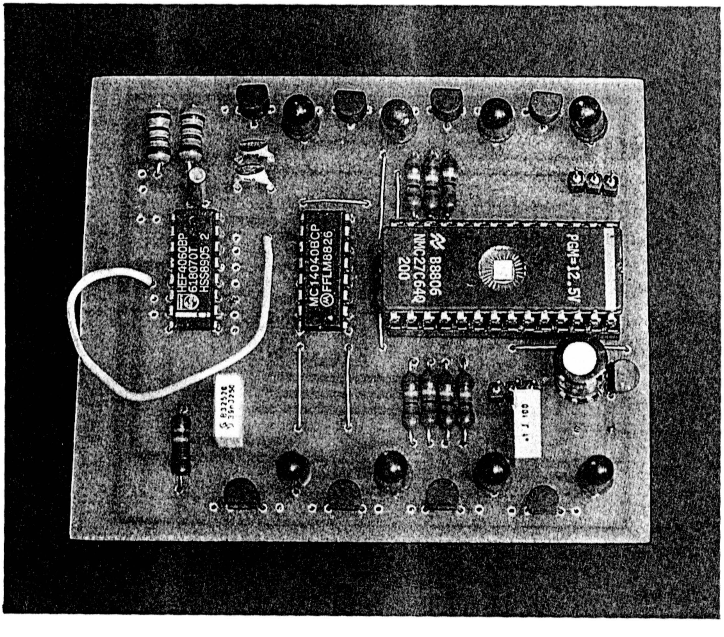
Condensateurs :
C1 : 150 pF,
C2 : 150 pF,
C3 : 39 nF,
C4 : 0,1 µF,
C5 : 220 µF, 16 V.

Transistors :
T1 à T8 : 2N 2222 ou BC 107.

Circuits intégrés :
CI1 : CD 4060B,
CI2 : CD 4040B,
CI3 : 27C64 à programmer,
CI4 : 78L05 (régulateur positif 5V).

Autres semiconducteurs :
8 diodes LED rouges.

Divers :
1 quartz 32,768 kHz,
2 cavaliers 2 positions,
1 support "tulipe" 28 broches.



UN MINI-AUTOMATE PROGRAMMABLE

Ce montage encore fort simple mais très universel est une amélioration du séquenceur précédemment décrit : sa vocation est toujours de "dérouler" un cycle enregistré dans une EPROM, mais avec une possibilité de prise en compte simplifiée d'évènements extérieurs.

Sans arriver à la même souplesse de programmation qu'un contrôleur à microprocesseur, ce petit automate peut résoudre élégamment et à très peu de frais une vaste palette de problèmes d'automatisation.

Dans notre ouvrage *ALARME ET SURVEILLANCE A DISTANCE,* paru dans la même collection, il nous a servi de base à la réalisation d'un simulateur de présence et d'un transmetteur téléphonique d'alarme, tandis que nous découvrirons dans un prochain chapitre comment le programmer pour lire des cartes à puce !

La figure 2.4 reproduit le schéma complet de notre carte, qui est cette fois équipée de quatre entrées (pour des contacts) et de quatre sorties (pour des relais, moteurs, électro-aimants, etc.) : l'expérience montre que c'est suffisant pour de nombreux usages, et il ne serait pas très difficile de multiplier cette capacité s'il le fallait.

Le coeur du montage est toujours une EPROM, dans laquelle le *programme* va être enregistré.

Il serait également possible de travailler, ne serait-ce que lors de la phase de mise au point du programme, avec une mémoire vive à pile au lithium ("ZEROPOWER") : le choix est laissé à l'initiative de l'utilisateur, notre carte acceptant les deux solutions.

Chaque mot de huit bits de la mémoire va à nouveau représenter pour nous un *pas de programme,* mais cette fois selon les conventions de la figure 2.5 :

- Les quatre premiers bits de chaque adresse (qui "pèsent", en décimal, 1,2,4, et 8) commandent chacun l'un des quatre transistors de sortie : en programmant 11110000 (15 en décimal), les quatre transistors seront passants.

- Les quatre bits restants (qui "valent" 16,32,64, et 128) forment un *masque* servant à interroger ou non les quatre entrées : un bit à 0 signifie que l'on ne s'intéresse pas à l'état de l'entrée considérée. Un bit à 1 signifie par contre que le passage au pas de programme suivant ne devra se faire que si l'entrée correspondante se trouve reliée à la masse.

00001111 (240 en décimal) signifie donc que les quatre transistors de sortie sont bloqués, et que le pas de programme suivant ne sera exécuté que lorsque les quatre entrées se trouveront mises ensemble à la masse (fonction ET).

Notons que des fonctions OU sur les entrées seraient faciles à réaliser par simple mise en parallèle des contacts concernés.

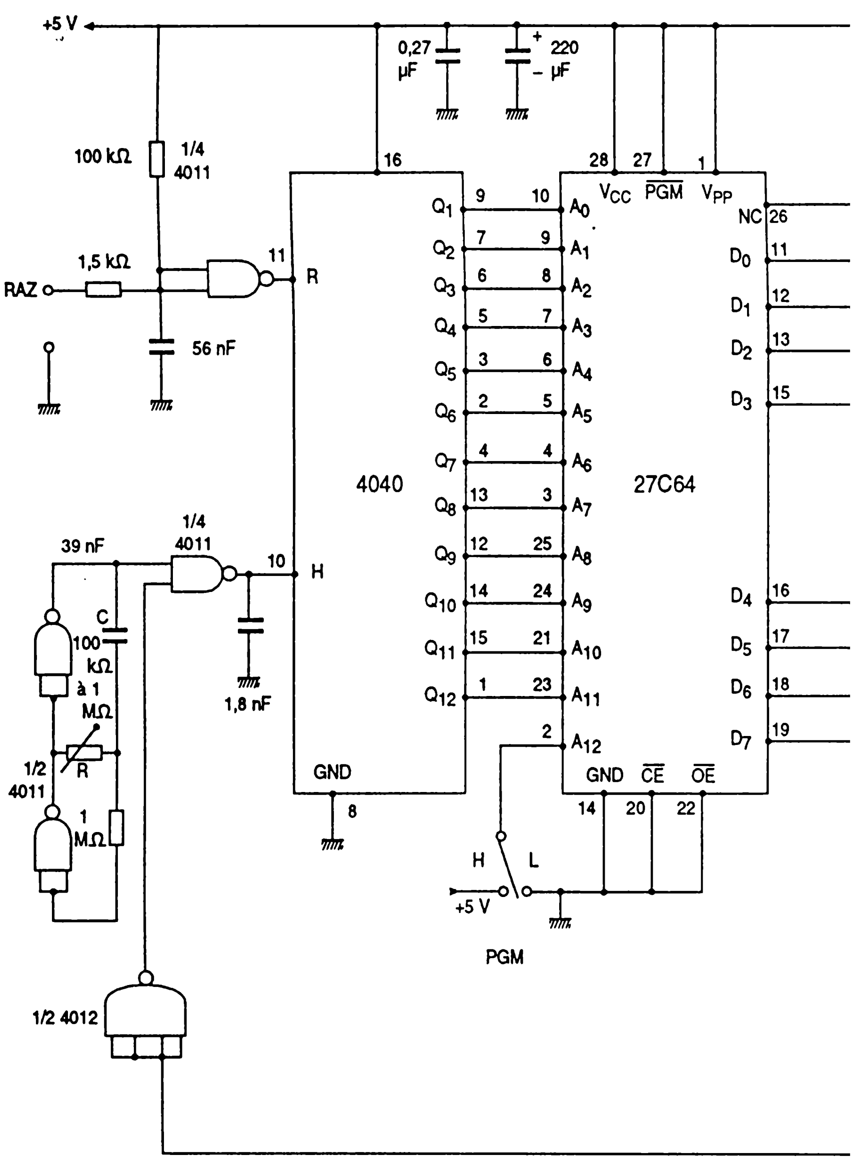
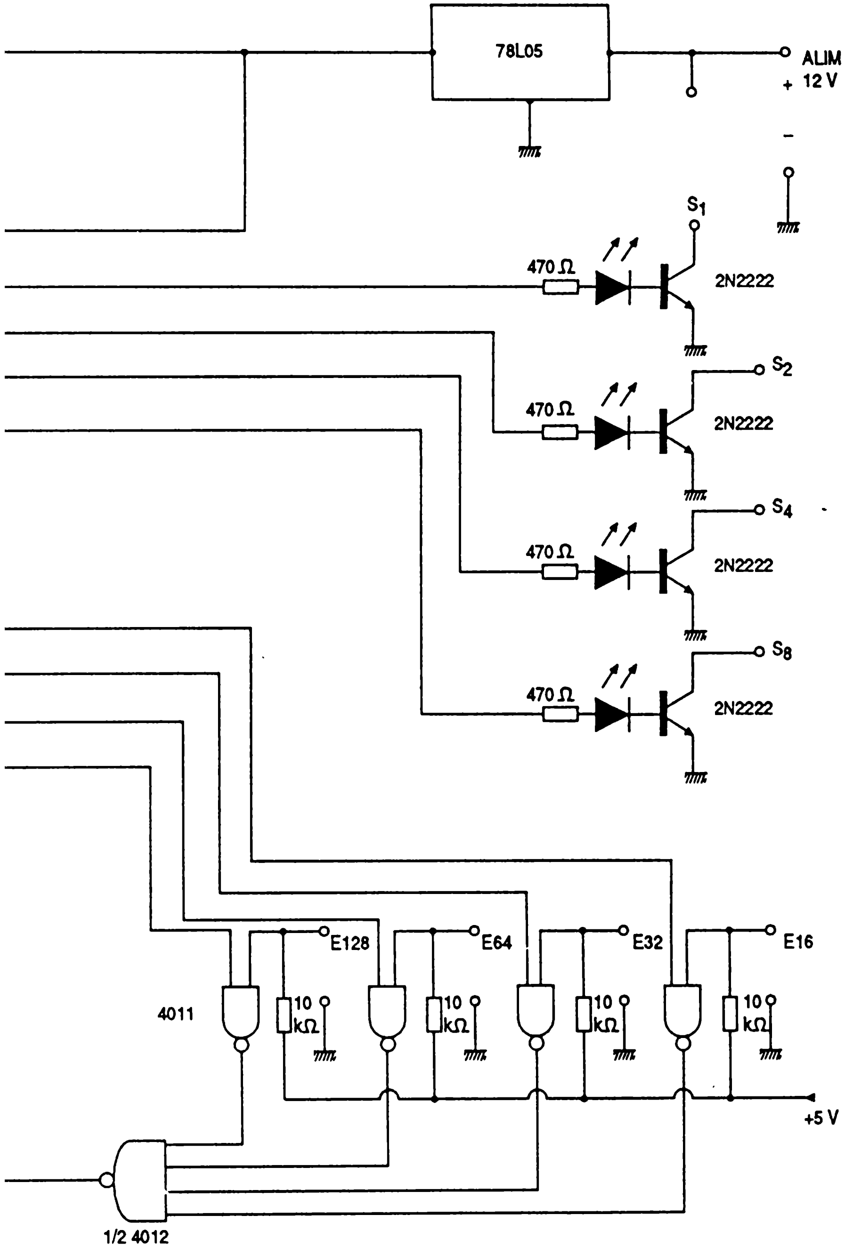


Fig. 2.4. – Schéma de l'automate.


78L05
ALIM
+ 12 V
-
S1
470 Ω
2N2222
S2
470 Ω
2N2222
S4
470 Ω
2N2222
S8
470 Ω
2N2222
E128
E64
E32
E16
4011
10 kΩ
10 kΩ
10 kΩ
10 kΩ
+5 V
1/2 4012

bit	0	1	2	3	4	5	6	7	
poids décimaux	1	2	4	8	16	32	64	128	poids décimaux
sorties	S_1	S_2	S_4	S_8	E_{16}	E_{32}	E_{64}	E_{128}	entrées
	1 = transistor passant 0 = transistor bloqué				1 = exécution du pas de programme suivant si entrée à la masse 0 = exécution du pas de programme suivant quel que soit l'état de cette entrée				

Fig. 2.5. – Principe de programmation de l'automate.

Ce comportement est facile à obtenir avec fort peu de matériel :

Un compteur binaire 4040 pilote directement les lignes d'adresse de la mémoire : remis à zéro par un réseau RC, il déclenche la lecture du premier pas de programme dès la mise sous tension de la carte. Il peut aussi être mis à zéro à tout moment en reliant l'entrée RAZ à la masse. Il peut parfois être intéressant de relier RAZ à l'une des sorties de la carte, afin que le programme se relance lui-même dans certaines circonstances (branchement conditionnel rudimentaire).

L'avance du compteur est confiée à un oscillateur d'horloge dont la période peut être réglée dans une très large plage par le choix d'une résistance et d'un condensateur : des temporisations multiples de cette période (environ 0,5 RC) pourront donc facilement être introduites dans le programme en enchaînant tout simplement le nombre voulu de pas identiques.

Le signal d'horloge n'est autorisé à agir sur le compteur que si l'état des entrées est conforme aux conditions programmées dans le "masque" du pas de programme en cours.

Sur le plan logique, il serait difficile de faire plus simple, mais cette structure est déjà très riche d'applications.

Sur le plan des interfaces, les sorties se font sur des transistors 2N 2222 montés en collecteur ouvert. Des voyants LED sont également prévus pour faciliter les simulations et les contrôles.

Du côté des entrées, des résistances de *tirage* sont prévues vers l'alimentation positive : on pourra donc indifféremment connecter de simples contacts entre entrées et masse, ou brancher des capteurs générant des niveaux TTL.

Un régulateur 5 V intégré sur la carte permet d'alimenter celle-ci sous n'importe quelle tension comprise approximativement entre 6 et 15 V.

Le câblage de ce montage est prévu sur un circuit imprimé simple face dont la figure 2.6 fournit le tracé. Tous les trous étant percés à 1 mm, il reste à implanter les composants selon le plan de la figure 2.7, sans oublier les quelques straps en fil rigide qui ont été nécessaires pour éviter le recours à la technique double face.

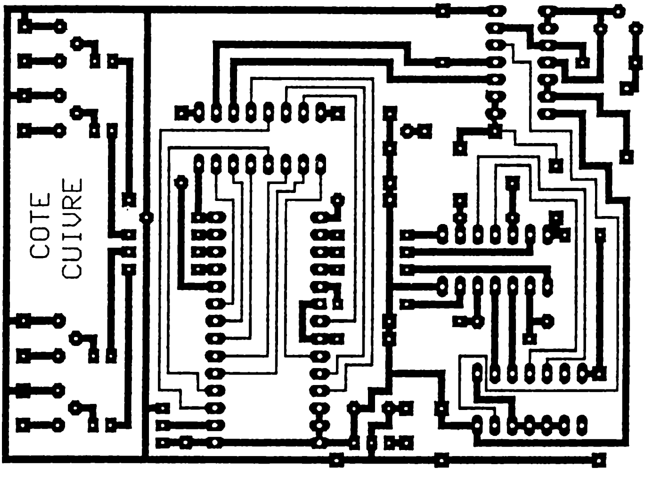


Fig. 2.6. – Circuit imprimé de l'automate.

Il est souhaitable d'utiliser un support de bonne qualité (type "tulipe") pour la mémoire, car la souplesse de programmation de cette carte laisse présager de nombreux changements de programme !

Un cavalier mobile permet de mettre en service l'une ou l'autre des deux moitiés de la mémoire, ce qui autorise l'enregistrement de deux programmes différents (par exemple un pour le jour et un pour la nuit).

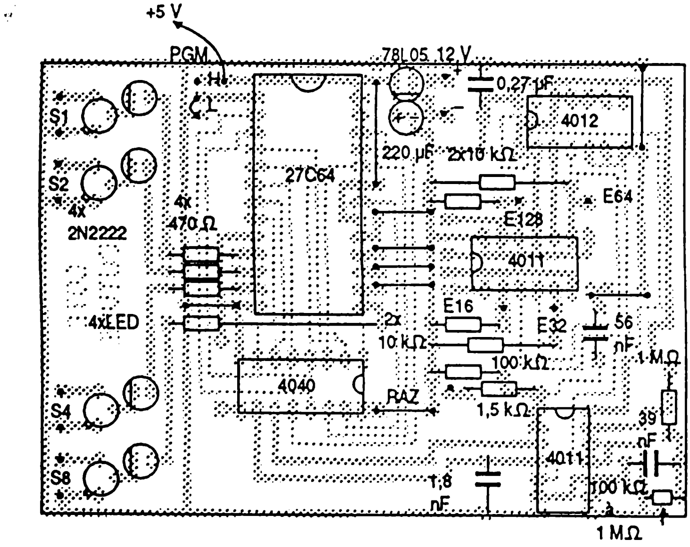


Fig. 2.7. – Plan de câblage de l'automate.

La capacité de la 27C64 utilisée (4096 pas de programme au total) suffit amplement pour un tel usage, et ne sera d'ailleurs que bien rarement utilisée en totalité.

Il n'y a à vrai dire aucun inconvénient à laisser une partie de la mémoire inoccupée, sauf peut-être lorsque l'on écrit un programme qui doit *boucler* en exploitant le fait que le compteur revient de lui-même à zéro lorsqu'il a *fait le tour* de toutes les adresses.

Il s'agit maintenant de programmer l'EPROM en fonction de l'application envisagée, car sans programme dans sa mémoire, le montage ne peut que rester parfaitement inerte : tout allumage d'un voyant trahirait une erreur de câblage, un composant défectueux, ou une mémoire mal effacée.

La *feuille de programmation* dont un modèle à photocopier est offert à la figure 2.8, permet de procéder à l'analyse dans la colonne de gauche (action) puis à la synthèse dans les colonnes de droite. Notons que la dernière colonne (octet décimal) n'est nécessaire que pour les utilisateurs d'un micro-ordinateur dont le programmateur d'EPROM attend les données à programmer sous cette forme : les utilisateurs d'un programmateur manuel muni d'un interrupteur par bit n'auront pas à la remplir, ce qui leur épargnera quelques additions !

PROGRAMME :

PAS N°	ACTION	SORTIES				ENTREES				OCTET
		1 S1	2 S2	4 S4	8 S8	16 E1	32 E2	64 E3	128 E4	décimal
1										
2										
3										
4										
5										
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										
21										
22										
23										
24										
25										

Fig. 2.8. – Feuille de programmation de l'automate.

Liste des composants du mini-automate

Résistances (5% 1/4 W) :
R1 : 1,5 kΩ,
R2 : 100 kΩ,
R3 : potentiomètre ajustable 100 kΩ à 1 MΩ,
R4 : 1 MΩ,
R5 à R8 : 470 Ω,
R9 à R12 : 10 kΩ.

Condensateurs :
C1 : 220 µF 16 V,
C2 : 0,27 µF,
C3 : 56 nF,
C4 : 39 nF,
C5 : 1,8 nF.

Transistors :
T1 à T4 : 2N 2222 ou BC 107.

Circuits intégrés :
CI1 et CI3 : CD 4011B,
CI2 : CD 4012B,
CI4 : CD 4040B,
CI5 : 27C64 à programmer,
CI6 : 78L05 (régulateur positif 5 V).

Autres semiconducteurs :
4 diodes LED rouges.

Divers :
1 support "tulipe" 28 broches.

UN "MICROCONTRÔLEUR" A Z80

Le terme *microcontrôleur* sert généralement à désigner des microprocesseurs "monochip" intégrant l'unité centrale, un générateur d'horloge, un certain nombre de ports d'entrée-sortie, et de la mémoire RAM et/ou ROM.

Lorsque la mémoire morte est réalisée en technologie EPROM, on se trouve en présence de véritables composants personnalisables, rassemblant un système micro-informatique complet dans un seul boîtier de circuit intégré.

Les avantages sont évidents en matière de miniaturisation et de simplification des circuits, mais il y a aussi quelques inconvénients.

La solution développée ici consiste à reconstituer l'équivalent d'un tel microcontrôleur sur une petite carte imprimée, autour d'un processeur Z80.

Nous allons voir que ce choix se révèle avantageux à plus d'un titre.

Les microcontrôleurs monochip à EPROM incorporée sont de superbes composants, qui permettent de développer des applications extrêmement performantes sous un très petit volume.

Sauf exécution "OTP" (versions non effaçables en boîtier plastique économique), l'indispensable grand boîtier céramique à fenêtre de quartz pèse cependant sur le prix de ces composants.

Par ailleurs, une EPROM est un composant relativement fragile, que l'on a vite fait d'endommager lors de multiples programmations et effacements : détruire une EPROM ordinaire n'est pas tragique, mais il en va tout autrement d'un coûteux microcontrôleur.

Ajoutons qu'un programmateur spécial ou au moins un adaptateur est indispensable pour le chargement des logiciels.

Enfin, les microcontrôleurs courants se programment dans un langage assembleur qui leur est propre : cela suppose un apprentissage, et souvent l'acquisition de logiciels de développement spécifiques.

De son côté, le Z80 n'est évidemment pas le dernier-né des microprocesseurs, mais il reste une excellente unité centrale qui a su conquérir ses lettres de noblesse.

Très bon marché, ce composant a équipé et équipe toujours de nombreux micro-ordinateurs économiques (AMSTRAD CPC notamment), pour lesquels existent d'innombrables logiciels de développement peu coûteux.

Cela n'empêche cependant nullement de faire de la mise au point de programmes Z80 sur PC et compatibles, car il existe de nombreux assembleurs conçus à cet effet.

C'est sur Z80 que beaucoup de développeurs se sont initiés à la programmation des microprocesseurs, et certains ne souhaitent pas repartir à zéro avec un autre langage assembleur.

Et curieusement, l'unification de l'Allemagne marque le début d'une nouvelle jeunesse pour le Z80 : ce processeur était depuis longtemps fabriqué illégalement à l'Est et le sera dorénavant de façon tout à fait officielle : il faut donc s'attendre à une possible chute des prix, et à un regain d'activité autour de ce composant en Europe, car la "Grande Allemagne" est maintenant riche d'excellents programmeurs de Z80.

Le schéma de notre carte, reproduit à la figure 2.9, fait appel à une architecture bien connue et, il est vrai, difficilement contournable : unité centrale, mémoire, tampons et verrous d'entrée-sortie, le tout desservi par les traditionnels bus d'adresses et de données.

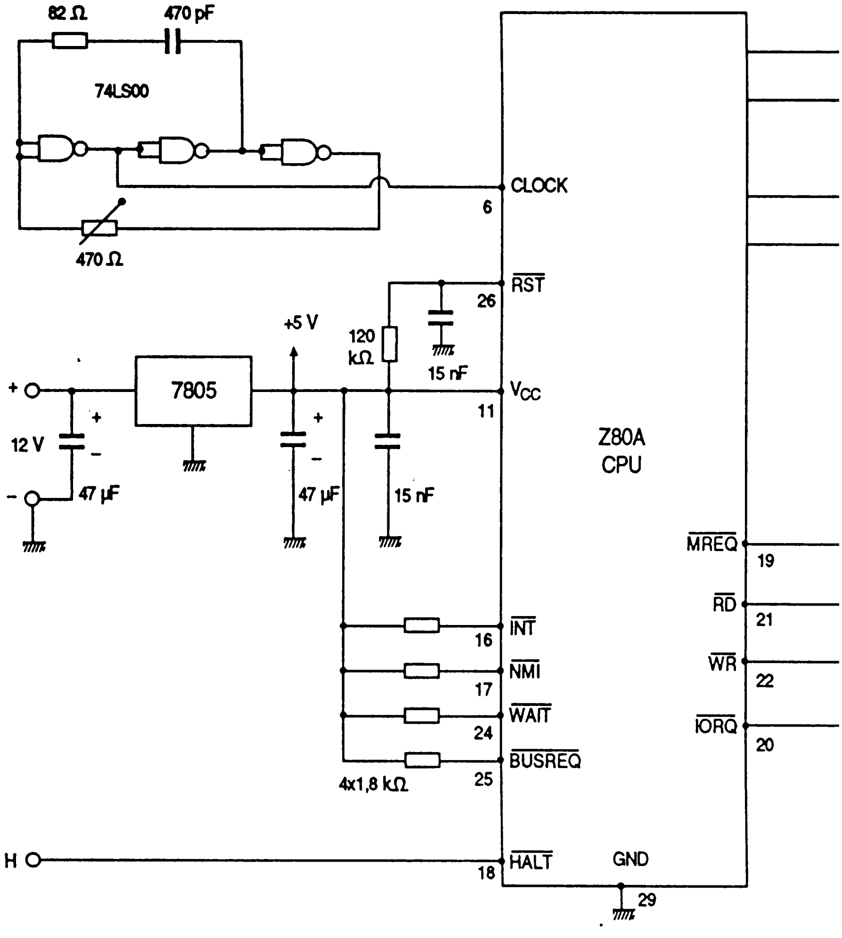
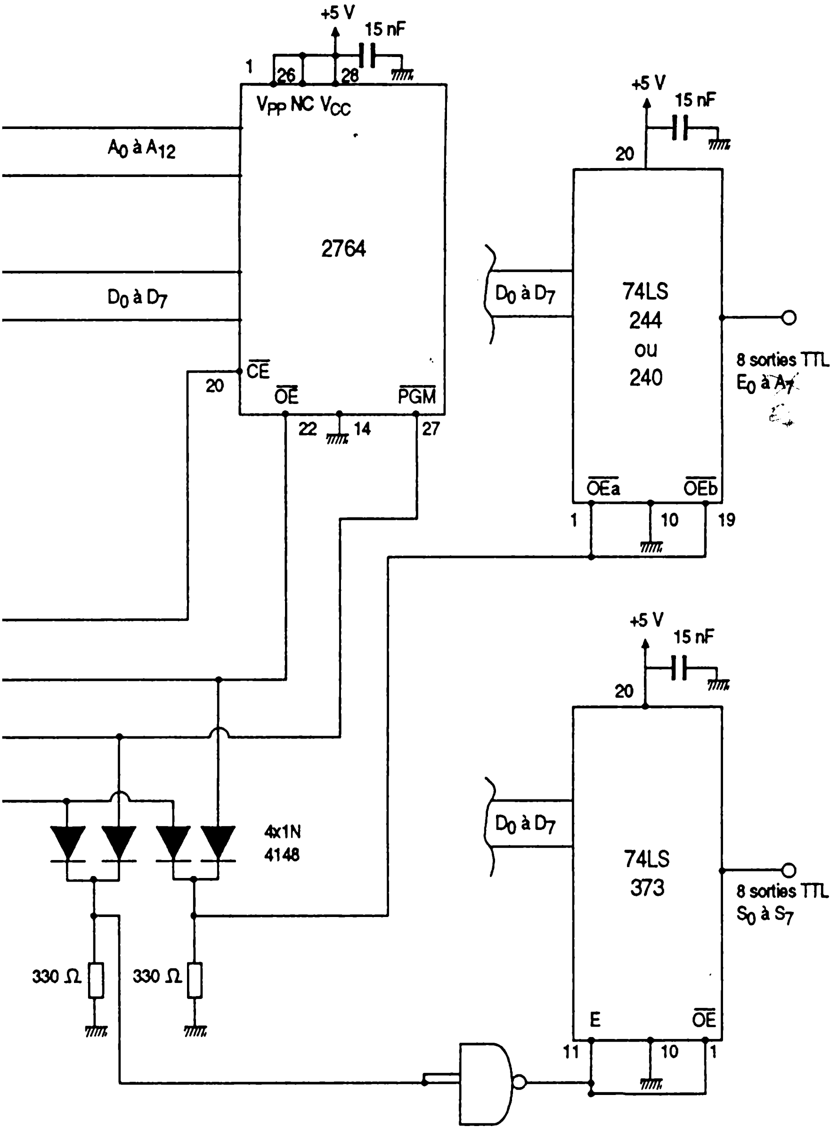


Fig. 2.9. – Schéma du microcontrôleur.


+5 V
15 nF
1
26
28
Vpp NC Vcc
A0 à A12
2764
D0 à D7
CE
20
OE
PGM
22
14
27
+5 V
15 nF
20
D0 à D7
74LS
244
ou
240
8 sorties TTL
OEa
OEb
1
10
19
+5 V
15 nF
20
4x1N
4148
D0 à D7
74LS
373
8 sorties TTL
S0 à S7
330 Ω
330 Ω
E
OE
11
10
1

A y regarder de plus près, on remarquera cependant un certain nombre d'originalités, pour la plupart simplificatrices et donc génératrices d'économies et de miniaturisation :

Pour commencer, le classique quartz d'horloge est absent : un oscillateur RC le remplace, dont la fréquence est ajustable par un potentiomètre, et modifiable dans une large mesure par le choix d'une valeur de condensateur.

Assez peu d'applications exigent en effet la précision du quartz, tandis qu'une possibilité d'ajustement manuel des temporisations d'un programme se révèle souvent fort utile.

Pas de mémoire RAM non plus, du moins à première vue : un seul support à 28 broches est prévu, qui peut accueillir une simple EPROM 2764 mais tout aussi bien, grâce à une ligne d'écriture supplémentaire, une RAM "ZEROPOWER" à pile lithium incorporée (MK 48Z08 de SGS-THOMSON, BQ 4010 de BENCHMARQ, DS1225 de DALLAS, etc.)

Avec la seule EPROM, il faut se plier à quelques restrictions de programmation et rentabiliser au maximum toute la panoplie de registres et d'instructions évoluées qui font la force du Z80. En particulier, l'absence de "pile machine" interdit l'usage de sous-programmes (instructions CALL et RET), et la sauvegarde des registres (instructions PUSH et POP).

La 2764 pouvant accueillir jusqu'à 8 K-octets de code machine, on peut toutefois largement se permettre d'alourdir ainsi la programmation.

L'emploi d'une RAM ZEROPOWER réunit les avantages des ROM et des RAM en un seul boîtier, sur la totalité des 8 K-octets du "plan mémoire" de la carte : tout ce qui n'est pas occupé par le programme est disponible comme de la RAM normale, dans laquelle on peut lire et écrire à volonté, mais qui ne perdra pas son contenu lors de la mise hors tension ou même du débrochage du boîtier : allez en faire autant avec seulement un monochip...

En phase de mise au point du logiciel, plus besoin non plus d'effacement puis de reprogrammation d'EPROM : n'importe quel octet de la mémoire peut être modifié séparément et instantanément sur l'ordinateur servant au développement.

Bien sûr, la ZEROPOWER coûte beaucoup plus cher qu'une EPROM de capacité équivalente, mais les services rendus ne sont pas du tout les mêmes. Rien n'empêche d'ailleurs de faire la mise au point dans une ZEROPOWER, puis de passer à une EPROM, voire à une OTP, lorsque le logiciel sera définitivement figé.

Les circuits de sélection sont également réduits à leur plus simple expression : comme il n'y a qu'un seul boîtier de mémoire, le signal /MREQ de

l'unité centrale peut servir directement à sa sélection, tandis que /RD et /WR peuvent rejoindre tout aussi directement ses broches de lecture et d'écriture.

Du côté des ports, nous avons prévu huit lignes d'entrée (un 74LS244) et huit lignes de sortie (un 74LS373) : il faut donc fabriquer deux signaux spécifiques à partir de /IORQ, /RD, et /WR. Deux portes à diodes suffisent, secondées par un inverseur constitué de la porte NAND inutilisée par le générateur d'horloge : on ne saurait faire plus compact, là où on trouve d'ordinaire plusieurs boîtiers TTL ou même un PAL spécialement programmé !

Nous aurions pu décider d'alimenter cette carte directement sous 5 V, et économiser ainsi encore quelques centimètres carrés. Pour des raisons pratiques, nous avons préféré ajouter un régulateur 7805 et quelques condensateurs, le tout permettant de se contenter de n'importe quelle source continue délivrant entre 7 et 30 V environ : l'utilisateur appréciera...

Compte tenu de la complexité des interconnexions nécessaires, la réalisation d'une carte simple face exigerait un nombre prohibitif de straps, que l'on ne pourrait guère réduire qu'en adoptant une disposition pas forcément très heureuse des composants.

Afin de privilégier la compacité et la commodité d'accès aux entrées-sorties, nous avons pris le parti de travailler en double face.

La disposition des composants et des connecteurs a d'abord été définie sans se soucier le moins du monde du tracé des pistes, et la délicate phase de *routage* a été confiée à un logiciel spécialisé : BOARDROUTER, routeur automatique anglais désormais disponible en France (voir notre ouvrage *CIRCUITS IMPRIMÉS* paru dans la même collection).

Quelques pistes "stratégiques" ont toutefois été routées manuellement, puis quelques optimisations ont été opérées sur les tracés proposés par le programme.

Il résulte de ce processus un nombre de *vias* (traversées entre faces) sûrement supérieur au minimum indispensable, mais on y gagne une disposition très pratique des connexions d'entrée-sortie et d'alimentation : en ligne sur un petit côté de la carte, et dans un ordre logique.

Cela permet de considérer la carte achevée comme un simple *composant* facile à insérer dans une réalisation plus complexe, mais dans lequel on peut échanger des mémoires ou remplacer des circuits intégrés endommagés !

Le tout pour un prix inférieur à celui d'un monochip, du moins si on ne fait pas intervenir le temps nécessaire aux opérations de câblage.

La figure 2.10 fournit le tracé des pistes du côté *soudures*, et la figure 2.11 celui du côté *composants*, dans l'ensemble perpendiculaire au précédent (sauf pistes routées manuellement).

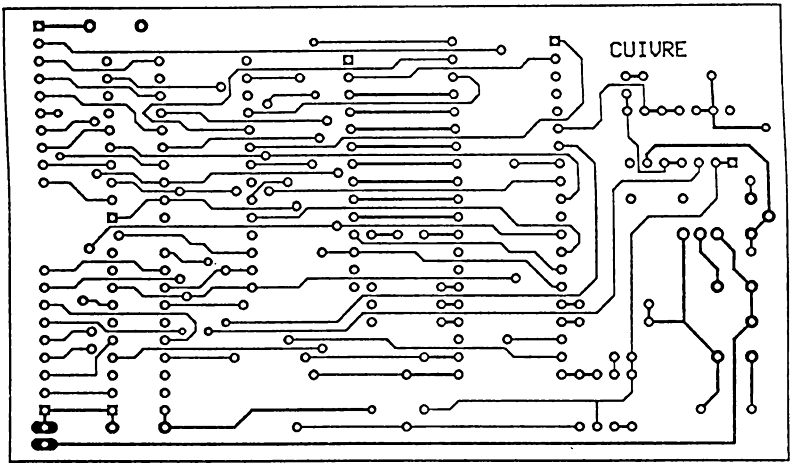


Fig. 2.10. – Côté soudures du circuit imprimé du microcontrôleur.

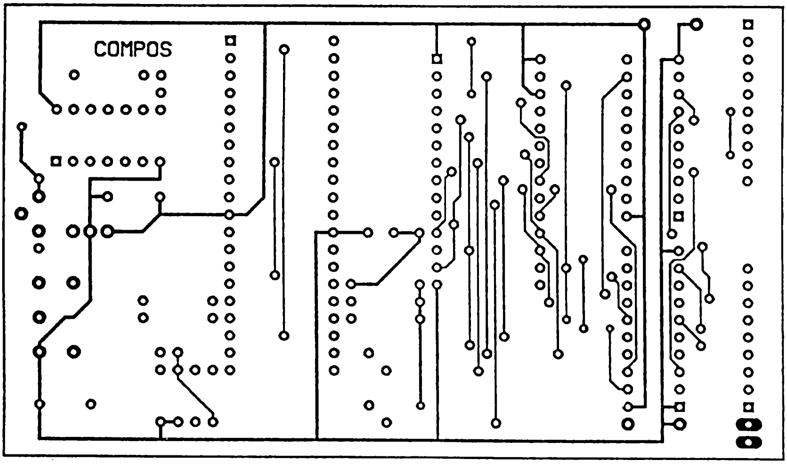


Fig. 2.11. – Côté composants du circuit imprimé du microcontrôleur.

Le plan de la figure 2.12 montre comment implanter les composants, qu'il faudra souder sur les deux faces lorsque des pistes partent des pastilles du côté composants.

Pour cette raison, nous conseillons vivement d'employer des *barrettes sécables tulipe* pour confectionner des supports pour tous les circuits intégrés : à défaut, les opérations de soudage seraient inutilement compliquées et d'éventuels remplacements de composants fort délicats.

A ce sujet, précisons d'ailleurs que le 74LS244 peut être remplacé à volonté par un 74LS240 lorsque l'application nécessite une complémentation des entrées : une entrée mise à la masse est considérée comme étant au niveau logique 0 par le LS244, mais au niveau 1 par le LS240.

Au pas de 2,54 mm, les pastilles du bord de la carte peuvent recevoir toutes sortes de connecteurs, le plus indiqué étant à notre avis des barrettes sécables de picots carrés, droits ou coudés.

Les opérations de câblage achevées, pas question évidemment de tester cette carte sans lui faire exécuter un programme !

Un très simple logiciel de sept octets suffit pour contrôler entièrement le montage :

IN A,(127)	219	127	
OUT (127),A	211	127	
JP 8192	195	0	32

Puisqu'il n'y a pas de décodage de numéro de port, l'adresse 127 est parfaitement arbitraire : en fait, n'importe quelle valeur décimale comprise entre 0 et 255 (00h à FFh en hexadécimal) ferait l'affaire, ce qui facilite grandement "l'importation" de logiciels mis au point sur des ordinateurs utilisant des adresses spécifiques.

Par ailleurs, l'instruction JP 8192, qui commande normalement l'exécution de l'instruction implantée à l'adresse décimale 8192 (2000h en hexa), déclenche en réalité le *bouclage* du programme à l'adresse zéro. Cela tout simplement parce que la ligne d'adresse A13 (de même que les suivantes) reste inutilisée, son *poids* décimal étant précisément 8192 (2 puissance 13).

Sera donc considérée comme valant zéro toute adresse pour laquelle les lignes A0 à A12 sont à zéro : il y a en fait sept "fantômes" de notre mémoire de 8 K-octets dans l'espace de 64 K adressable par le Z80.

Là encore, cette particularité facilite l'utilisation directe de programmes assemblés à des adresses différentes de zéro, sur des machines équipées de Z80.

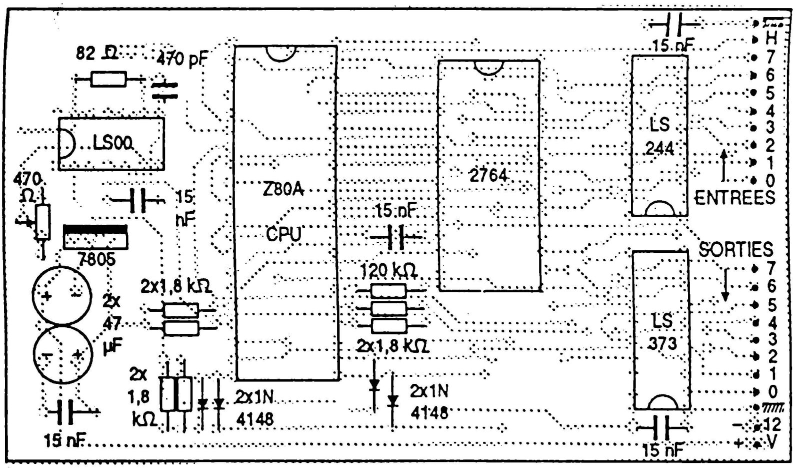
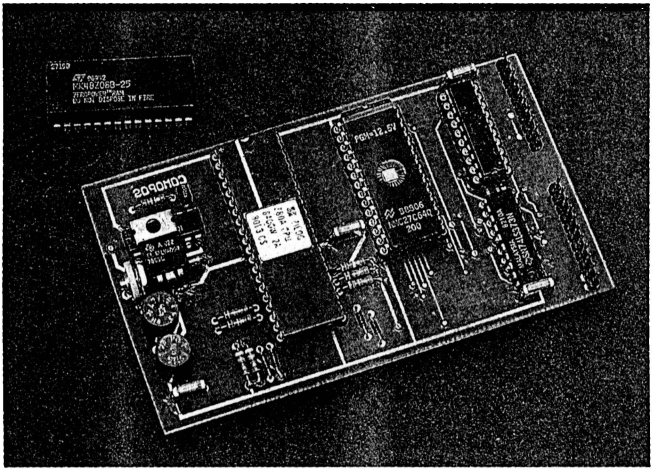


Fig. 2.12. – Plan de câblage du microcontrôleur.



Notre petit programme recopie en permanence les huit entrées sur les huit sorties : pour vérifier entièrement la carte, il suffit donc de mettre tour à tour à la masse chacune des huit entrées en s'assurant que la sortie de même numéro (et elle seule) passe simultanément au niveau bas.

En principe, des résistances *de tirage au positif* sont nécessaires pour interfacer ainsi des contacts avec les entrées du 74LS244. On peut s'en passer à l'occasion de ce simple essai, et dans le cas où la carte reçoit de véritables signaux TTL : nous ne les avons donc pas prévues sur la carte.

Si on utilise une mémoire ZEROPOWER, le même logiciel permet de procéder à un test semblable, mais il faut le compliquer un peu si on veut s'assurer au passage du bon fonctionnement de la partie RAM, et donc notamment de la fameuse *pile machine* :

IN A,(127)	219	127	
CALL 9	205	9	0
JP 0	195	0	0
NOP	0		
OUT (127),A	211	127	
RET	201		

Si tout va bien, le résultat obtenu doit être le même que précédemment. Par contre, si on tente de charger ce logiciel dans une simple EPROM, un blocage se produira immédiatement, les sorties prenant définitivement l'état qui était celui des entrées lors de la mise sous tension : en effet, faute de *pile machine* en RAM, l'instruction RET ne dispose pas de l'adresse de retour au programme appelant !

Pour la commodité de ces essais (et de ceux à venir avec vos propres programmes !), nous fournissons à la figure 2.13 le schéma d'un très simple adaptateur muni de huit interrupteurs DIL, de huit voyants LED, et d'une diode de protection contre les inversions de polarité d'alimentation.

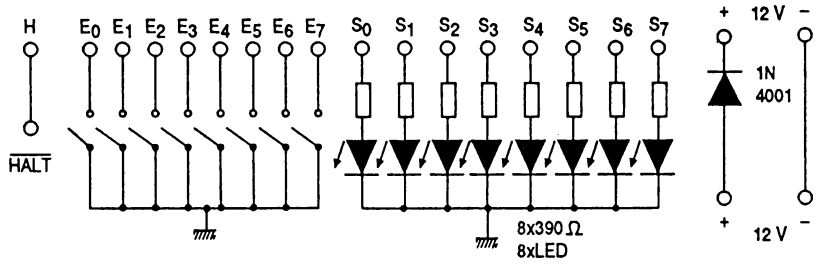


Fig. 2.13. – Schéma de la carte de test.

Deux douilles de 4 mm permettent le raccordement facile à une alimentation de laboratoire, tandis que les pastilles servant au raccordement à la carte "microcontrôleur" sont disposées exactement en regard des connecteurs de celle-ci.

Le circuit imprimé de la figure 2.14 possède une dimension en commun avec celui du microcontrôleur, et peut cette fois être gravé en simple face.

Le plan de la figure 2.15 montre comment implanter les quelques composants nécessaires, les diodes LED devant être du type *miniature*.

Une douille "tulipe" est prévue pour l'accès à la sortie /HALT du Z80, qu'il peut être utile de tester lors de l'exécution de certains programmes utilisant l'instruction du même nom (arrêt de l'unité centrale).

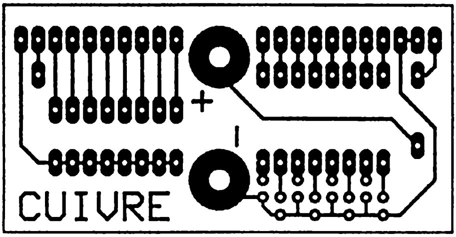


Fig. 2.14. – Circuit imprimé de la carte de test.

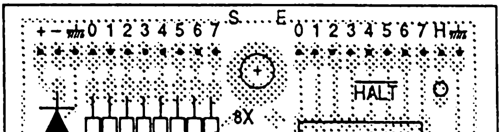


Fig. 2.15. – Plan de câblage de la carte de test.

Voici donc une carte de dimensions très réduites, facile à réaliser en autant d'exemplaires que nécessaire au moyen de composants très courants et peu coûteux.

Elle n'en offre pas moins un confort d'utilisation certain, du fait de la disposition commode de ses connecteurs de raccordement au "monde extérieur", et de la souplesse de son organisation mémoire.

Des atouts qui en font une solution à considérer de près pour toutes sortes de projets d'automatisation, pour lesquels on appréciera certainement de pouvoir la programmer dans un langage aussi *populaire* que l'assembleur Z80.

Liste des composants du microcontrôleur à Z80 :

Résistances (5% 1/4 W)
R1 : 82 Ω
R2 : potentiomètre ajustable 470 Ω
R3 à R8 : 1,8 kΩ
R9 : 120 kΩ

Condensateurs :
C1 et C2 : 47 µF 16 V radial
C3 : 470 pF
C4 à C8 : 15 nF

Circuits intégrés :
CI1 : Z80A CPU
CI2 : 27C64 (ou 2764) à programmer
CI3 : 74LS244
CI4 : 74LS373
CI5 : 7805 (régulateur positif 5 V)
CI6 : 74LS00

Autres semiconducteurs :
D1 à D4 : 1N 4148

Divers :
barrettes sécables "tulipe"
barrettes sécables à picots carrés

③ SYNTHÉTISEURS DE SON PERSONNALISABLES

Plusieurs techniques radicalement différentes sont à la disposition du développeur souhaitant donner la parole à ses circuits électroniques. La plus évidente consiste à numériser un son naturel à l'aide d'un convertisseur analogique-numérique (CAN), à le stocker dans une mémoire, puis à le reproduire par l'intermédiaire d'un convertisseur numérique-analogique (CNA).

L'inconvénient de la méthode (dite PCM directe) est que pour obtenir la simple qualité téléphonique (bande passante de 4 kHz), il faut au minimum 8 K-octets de mémoire par seconde de son ! En effet, le théorème de Shannon énonce que la fréquence d'échantillonnage doit être au moins double de celle du signal à digitaliser, tandis qu'un codage de l'amplitude sur huit bits peut être considéré comme un minimum pour garantir une résolution satisfaisante (il faut 10 à 12 bits pour atteindre une qualité véritablement excellente).

Ce principe ne peut donc guère convenir qu'à des applications simples, limitées à l'émission de sons courts, mais pas forcément cantonnées à de la parole : on peut déjà en trouver beaucoup, tandis que les capacités des mémoires disponibles ne cessent de croître malgré des prix en baisse...

Une toute autre approche consiste à découper la parole en un nombre limité de composantes élémentaires : les formants, communs à toutes les langues, ou les phonèmes, très différents par exemple entre l'anglais et le français. Préprogrammées en mémoire morte dans des circuits intégrés spécialisés, ces *briques* peuvent être appelées à volonté en envoyant sur un bus les codes binaires appropriés. On peut ainsi convertir en parole un texte venant de la prise d'imprimante d'un ordinateur, mais guère reconstituer de façon convenable un aboiement de chien ou un coup de feu.

Malgré les considérables efforts qu'ont nécessité leur mise au point, ces techniques sont en voie d'abandon car d'autres méthodes existent maintenant, qui permettent de limiter la taille mémoire nécessaire pour stocker des sons naturels numérisés. Elles dérivent souvent de celles employées en téléphonie numérique, ce qui est précisément le cas de la modulation delta.

Comme son nom l'indique presque, le principe de la modulation delta consiste à enregistrer ou transmettre à chaque instant, non plus la valeur instantanée du signal mais le sens de sa variation par rapport à l'instant d'échantillonnage précédent, soit un seul bit au lieu de huit.

Le gain en volume d'informations est évidemment considérable, bien que la figure 3.1 permette de constater que les résultats obtenus peuvent être assez honorables pourvu que les paramètres de conversion soient choisis avec soin.

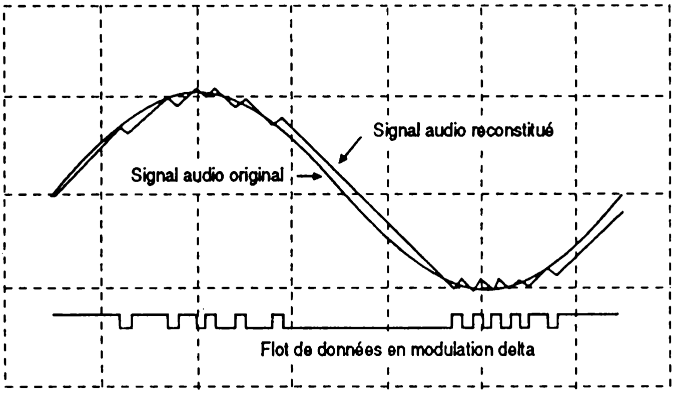


Fig. 3.1. – Principe de la modulation Delta (doc. Motorola).

Le signal analogique est appliqué à un comparateur, dont la sortie échantillonnée à un rythme fixe (horloge) constitue l'information digitalisée. Selon sa valeur (un ou zéro), celle-ci détermine la polarité (croissante ou décroissante) de la rampe d'un intégrateur dont la sortie attaque la seconde entrée du comparateur.

Cette boucle fermée se trouve donc asservie au signal d'entrée, et on retrouve en sortie de l'intégrateur un signal analogique peu différent de celui de départ.

Si après transmission ou stockage, le message digital série disponible en sortie de l'échantillonneur est appliqué à un intégrateur identique, alors la sortie de celui-ci délivrera un signal analogique raisonnablement fidèle à l'original.

L'amélioration connue sous le nom de CVSD (*Continuously Variable Slope Delta modulation*) consiste à adapter la pente de l'intégrateur (c'est à dire la vitesse de croissance ou de décroissance de sa tension de sortie) en fonction de l'évolution du message numérique : on pourra par exemple rendre l'intégrateur plus rapide si les trois ou quatre derniers bits sont identiques, mais différents algorithmes sont possibles à ce niveau.

Ce genre d'artifice accroît la dynamique, normalement très limitée, de la modulation delta et améliore donc le rapport signal à bruit.

Bien entendu, la mise en oeuvre pratique de ce procédé suppose des traitements supplémentaires des signaux analogiques avant modulation et après démodulation : limitation de niveau et filtrages fixant précisément la bande passante, en accord avec la fréquence d'échantillonnage choisie.

La modulation delta est à la base de nombreuses applications spectaculaires : téléphone numérique, réverbération digitale, truquage de voix, cryptage de conversations, et bien entendu magnétophones statiques.

UN DIGITALISEUR DE SON POUR PC ET COMPATIBLES

Il suffit d'un matériel fort simple pour faire acquérir à un compatible PC des sons de courte durée et de qualité acceptable : typiquement un convertisseur analogique-numérique 8 bits assez ordinaire et quelques composants permettant de l'interfacer, par exemple, à sa prise d'imprimante parallèle.

Une fois sauvegardé sur disque dur ou disquette, le son ainsi digitalisé pourra être ré-écouté à l'aide d'un montage encore plus simple (convertisseur numérique-analogique), ou transféré dans une EPROM pour reproduction à la demande par un montage autonome.

Typiquement, une 2764 peut ainsi héberger environ une seconde de son de qualité raisonnable : c'est suffisant pour réaliser des montages assez étonnants tels que sirènes parlantes (au voleur, au secours, etc.), des chiens électroniques aux aboiements fort réalistes, ou des effaroucheurs d'oiseaux imitant un cri de détresse approprié, voire même des appeaux électroniques pour la chasse.

Et bien entendu, le même principe pourrait être étendu à des EPROM d'un ou plusieurs mégabits, augmentant d'autant la durée des sons reproductibles.

Un exemple de son ainsi digitalisé est fourni sous la forme d'un listing, dans l'annexe placée en fin d'ouvrage.

Parmi les convertisseurs analogique-numérique les plus répandus et les moins chers, nous avons retenu l'ADC 0809 de National Semiconductor. Le schéma de la figure 3.2 montre que nous avons réuni ensemble ses huit entrées, et équipé le tout d'un circuit d'interface spécifique pour signaux audio : deux résistances polarisant l'entrée à la moitié de la tension d'alimentation (référence de conversion), deux diodes servant à écrêter les éventuels dépassements d'amplitude, un condensateur de liaison et une résistance de charge pour la source audio.

Celle-ci sera la prise casque ou HP supplémentaire d'un magnétophone à bobines disposant d'au moins deux vitesses, et si possible de quatre : pour sa part, l'auteur opère régulièrement sur un fidèle UHER REPORT 4000 vieux d'une vingtaine d'années mais offrant toujours la qualité studio.

Il est très important, pour la qualité du son reproduit, que le niveau d'attaque du convertisseur soit aussi proche que possible de 5 V crête à crête, c'est à dire 2,5 V crête ou 1,77 V efficace : trop fort, le signal serait écrêté et donc distordu, trop faible, il serait entaché de bruit de fond et aussi distordu, à cause du sous-emploi des 256 niveaux disponibles.

Il sera donc souhaitable de régler le niveau de lecture du magnétophone sous le contrôle d'un oscilloscope.

Un oscillateur à portes CMOS fournit le signal d'horloge nécessaire au convertisseur, tandis qu'un sélecteur 4512 réalise l'aiguillage de ses huit lignes de données, chacune à son tour, vers l'entrée /ACK de la prise CENTRONICS de l'ordinateur.

Nous retrouvons là un choix qui est cher à l'auteur depuis qu'il développe des extensions pour PC : en effet, cette prise pratiquement toujours disponible peut servir à la fois d'entrée et de sortie tandis que son utilisation n'exige aucune manipulation hasardeuse sur les *slots* et présente donc une grande sécurité pour la machine. Enfin, son brochage, ses signaux, et ses adresses d'accès sont bien standardisés.

La réalisation pratique est prévue sur le petit circuit imprimé simple face dessiné à la figure 3.3. L'implantation de la figure 3.4 accueille tous les composants nécessaires, à l'exception de la prise CENTRONICS qui sera montée au bout d'un morceau de câble plat à 9 conducteurs.

L'alimentation doit se faire sous 5 V, tension facile à prélever sur un connecteur quelconque de l'ordinateur ou à produire extérieurement. Le cas échéant, une simple pile de 4,5 V pourrait même faire l'affaire.

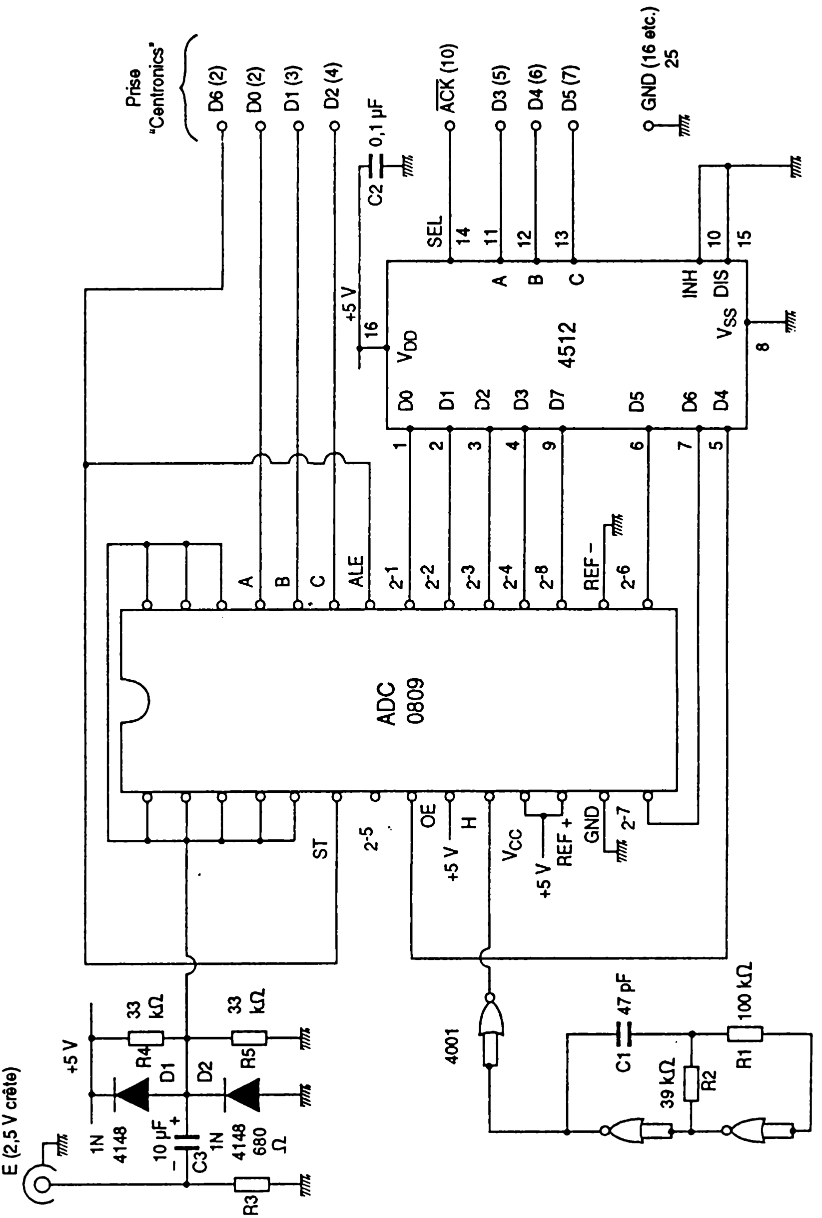


Fig. 3.2. – Schéma du digitaliseur de son.

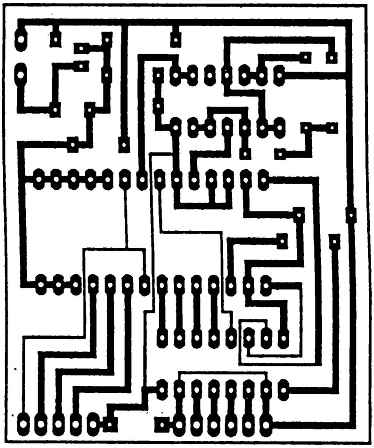


Fig. 3.3. – Circuit imprimé du digitaliseur de son.

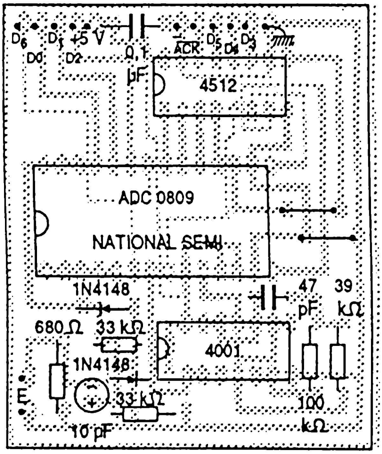
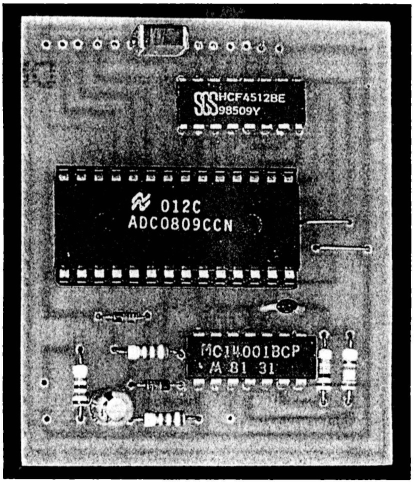


Fig. 3.4. – Plan de câblage du digitaliseur de son.



Liste des composants du digitaliseur de son :

Résistances (5% 1/4 W) :
R1 : 100 kΩ,
R2 : 39 kΩ,
R3 : 680 Ω,
R4 et R5 : 33 kΩ.

Condensateurs :
C1 : 47 pF,
C2 : 0,1 µF,
C3 : 10 µF 16 V radial.

Circuits intégrés :
CI1 : CD 4001B,
CI2 : ADC 0809 (NS),
CI3 : CD 4512B.

Autres semiconducteurs :
D1 et D2 : 1N 4148.

Un logiciel spécifique est évidemment nécessaire pour piloter le convertisseur, puis pour emmagasiner les données collectées dans un fichier disque.

Compte tenu de la brièveté du son à digitaliser, nous avons prévu de lire la bande au quart de la vitesse ayant servi à l'enregistrer, quitte à passer par une copie intermédiaire à demi-vitesse si le magnétophone utilisé ne possède que deux allures.

En relisant ainsi à 4,75 cm/s une bande enregistrée à 19 cm/s, la durée de l'opération de digitalisation augmente suffisamment pour que l'on puisse travailler confortablement, notamment au démarrage (utiliser la touche de pause du magnétophone). Malgré cet artifice, le logiciel doit être rapide : pas question donc de programmer en BASIC !

Pas tout à fait aussi rapide que l'assembleur mais infiniment plus que le BASIC, le Turbo-Pascal se prête particulièrement bien à ce genre de travail : le listing de la figure 3.5 devra donc être saisi puis traduit en un fichier exécutable, à l'aide du fameux compilateur de Borland.

```
program enregist;
uses crt;
var t:array[1..32000] of integer;
    f,g,h,j,k:integer;
    i:real;
    a:text;
begin
clrscr;
writeln('presser ENTER au démarrage du son');
readln;
k:=0;
for f:=1 to 4000 do
begin
port[888]:=0;
port[888]:=64;
```

(suite au verso)

Fig. 3.5. – Le logiciel de prise de son (version 4,77 MHz).

```
for g:=0 to 7 do
begin
port[888]:=8*g;
k:=k+1;
t[k]:=port[889];
end;
end;
clrscr;
writeln('----- TRAITEMENT EN COURS -----');
assign(a,'son.rom');
rewrite(a);
for f:=0 to 42 do
begin
g:=3*f;
write(a,' ',g,' ');
end;
write(a,' ',126,' ',127,' ',127,' ',128,' ',128,' ');
k:=0;
for f:=1 to 4000 do
begin
i:=0;
for g:=0 to 7 do
begin
k:=k+1;
if(t[k] and 64)=64 then i:=i+(exp((7-g)*ln(2)));
end;
j:=round(i);
write(a,' ',j,' ');
end;
write(a,' ',128,' ',128,' ',127,' ',127,' ',126,' ');
for f:=0 to 42 do
begin
g:=3*(42-f);
write(a,' ',g,' ');
end;
clrscr;
writeln('----- FICHIER  SON.ROM  DISPONIBLE -----');
close(a);
end.
(* COPYRIGHT 1991 Patrick GUEULLE*)
```

Fig. 3.5. – Le logiciel de prise de son (version 4,77 MHz).

Prévu pour tourner sur un PC de base fonctionnant à 4,77 MHz, on lui substituera la variante un peu ralentie de la figure 3.6 en présence d'un système cadencé à 8 MHz, et on pourrait s'inspirer de la technique de ralentissement utilisée pour adapter le programme à une machine encore plus rapide.

Il faudra veiller au bon synchronisme entre le magnétophone et le convertisseur : il est impératif que la durée du son (ralenti quatre fois) soit égale ou légèrement inférieure au temps qui s'écoule entre l'appui sur ENTER et l'apparition du message *traitement en cours*. Si tel n'était pas le cas, il faudrait soit recommencer la prise de son, soit ajuster la vitesse du logiciel. Quelques chronométrages sont donc à prévoir !

```
program rec;
uses crt;
var t:array[1..32000] of integer;
    f,g,h,j,k:integer;
    i:real;
    a:text;
begin
clrscr;
writeln('presser ENTER au démarrage du son');
readln;
k:=0;
for f:=1 to 4000 do
begin
port[888]:=0;
port[888]:=64;
for g:=0 to 7 do
begin
for h:=0 to 2 do
begin
port[888]:=8*g;
end;
k:=k+1;
t[k]:=port[889];
end;
end;
clrscr;
writeln('----- TRAITEMENT EN COURS -----');
assign(a,'son.rom');
rewrite(a);
for f:=0 to 42 do
begin
g:=3*f;
write(a,' ',g,' ');
end;
write(a,' ',126,'  ',127,'  ',127,'  ',128,'  ',128,' ');
k:=0;
for f:=1 to 4000 do
begin
i:=0;
for g:=0 to 7 do
begin
k:=k+1;
if(t[k] and 64)=64 then i:=i+(exp((7-g)*ln(2)));
end;
j:=round(i);
write(a,' ',j,' ');
end;
write(a,' ',128,'  ',128,'  ',127,'  ',127,'  ',126,' ');
for f:=0 to 42 do
begin
g:=3*(42-f);
write(a,' ',g,' ');
end;
clrscr;
writeln('----- FICHIER  SON.ROM  DISPONIBLE -----');
close(a);
end.
(* COPYRIGHT 1991 Patrick GUEULLE*)
```

Fig. 3.6. – Le logiciel de prise de son (version 8 MHz).

Si tout est conforme, il ne reste qu'à effectuer la digitalisation pour de bon : lancer le programme et mettre le magnétophone en lecture et pause, le début du son étant exactement positionné sur la tête de lecture.

Relâcher alors la touche de pause tout en appuyant simultanément sur ENTER de l'ordinateur. Arrêter complètement le magnétophone dès l'apparition du message "traitement en cours", et attendre celle de "Fichier SON.ROM disponible".

Même en Turbo-Pascal, il n'y a pas de temps à perdre : les données provenant du convertisseur sont donc entassées telles qu'elles arrivent dans un grand tableau mémoire, à raison d'un octet par bit, puis traitées une fois l'acquisition terminée.

Les bits reçus en série sur le fil /ACK sont ainsi réassemblés en 4000 octets, encadrés par deux séquences destinées à adoucir les transitions au début et à la fin de la reproduction du son (élimination des *clocks* de commutation).

Au terme de ce traitement qui prend tout de même un certain temps, le fichier SON-ROM est disponible sur le disque, représentant en principe une demi-seconde de son échantillonné à 8 kHz (soit 4096 octets, c'est-à-dire l'équivalent de la moitié d'une 27C64).

Il s'agit d'une suite de valeurs décimales, forme convenant au programmateur décrit dans notre ouvrage *COMPOSANTS ELECTRONIQUES PROGRAMMABLES*, paru dans cette même collection.

En présence d'un programmateur attendant plutôt un fichier binaire il faudra passer par l'intermédiaire de l'utilitaire ROMTOBIN.BAS, petit programme BASIC listé à la figure 3.7.

Un premier son pourra être programmé en maintenant la broche 2 de l'EPROM à la masse, et un second en la portant à +5 V.

Mais avant de réaliser le montage chargé de la reproduction du son à partir d'une EPROM, on souhaitera sans doute procéder à une écoute de contrôle de la façon la plus simple possible.

```
10 REM ---- ROMTOBIN ----
20 CLS:PRINT"Nom du fichier .ROM à convertir en .BIN ?"
30 INPUT R$:B$=R$+".BIN":R$=R$+".ROM"
40 OPEN R$ FOR INPUT AS #1
50 OPEN "R",#2,B$,1
60 CLS:PRINT"---- CONVERSION DE FORMAT EN COURS ----"
70 FIELD#2,1 AS A$
80 IF EOF(1) THEN 120
90 INPUT#1,M:M$=CHR$(M)
100 RSET A$=M$:PUT#2,F+1
110 F=F+1:GOTO 80
120 CLS:PRINT R$;" est transformé en ":B$
130 END
140 REM (c)1992 Patrick GUEULLE
```

Fig. 3.7. – Le logiciel de transcodage de fichiers.

UN REPRODUCTEUR DE CONTROLE POUR PC

Le petit montage de la figure 3.8 permet de lire le fichier dans n'importe quel ampli BF (chaîne HIFI par exemple) relié à la prise d'imprimante (CENTRONICS) du PC. C'est en fait une réplique du convertisseur numérique-analogique du reproducteur à EPROM qui sera décrit plus loin.

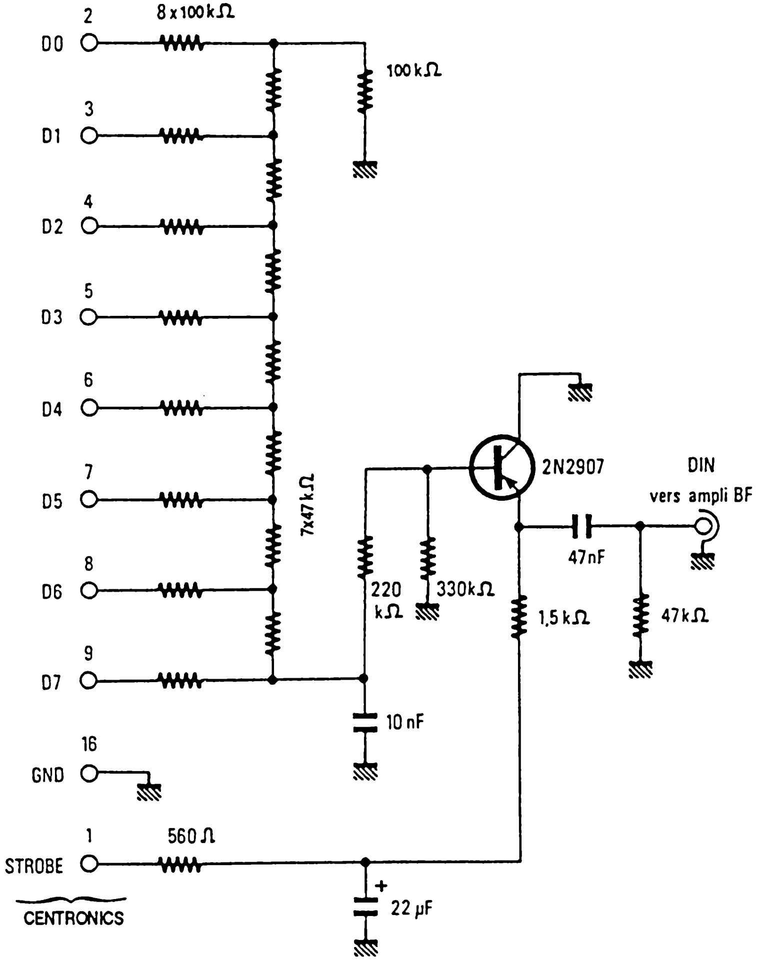


Fig. 3.8. – Schéma du reproducteur de contrôle.

Le câblage du circuit imprimé de la figure 3.9 se fera d'après le plan de la figure 3.10, qui prévoit l'implantation directe d'une prise DIN au brochage normalisé mono-stéréo : l'ampli pourra donc être raccordé à l'aide de tout cordon DIN déjà disponible.

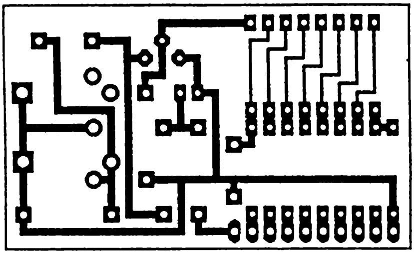


Fig. 3.9. – Circuit imprimé du reproducteur de contrôle.

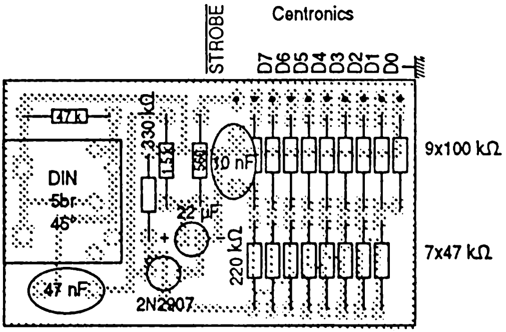


Fig. 3.10. – Plan de câblage du reproducteur de contrôle.

L'opération de lecture est confiée au logiciel de la figure 3.11, également écrit en *TURBO PASCAL* pour les mêmes raisons que précédemment, lesquelles obligent encore une fois à passer par l'intermédiaire d'un tableau en mémoire accessible plus rapidement que le fichier disque.

La boucle for j :=0 to 5 do permet d'ajuster la vitesse de lecture si nécessaire, la variable j ayant le même rôle que le sélecteur de vitesse d'un magnétophone ou d'un tourne-disque.

```
program lecture;
uses crt;
var m:array[0..4096] of integer;
    h:string[12];
    f:integer;
    e:boolean;
    g:integer;
    i:integer;
    j:integer;
    a:text;
begin
clrscr;
writeln('nom du fichier à lire ?');
readln(h);
h:=h+'.rom';
assign(a,h);
reset(a);
f:=0;
repeat
read(a,m[f]);
e:=seekeof(a);
f:=f+1;
until e=true;
close(a);
for i:=1 to 20 do
begin
for g:=1 to f do
begin
port[888]:=m[g];
for j:=0 to 5 do
begin
end;
end;
readln;
end;
end.
(* COPYRIGHT 1990 Patrick GUEULLE *)
```

Fig. 3.11. – Le logiciel de lecture.

Dans la variante de la figure 3.12, cette vitesse peut être réglée pendant l'exécution et de plus, il est possible de ne lire qu'une partie du fichier en spécifiant une adresse de début et une adresse de fin. Il devient ainsi envisageable d'isoler des parties de son que l'on souhaite prélever ou au contraire éliminer s'il s'agit de bruits parasites.

```
program lit;
uses crt;
var m:array[0..28672] of integer;
    h:string[12];
    f,g,i,j,ad,af,v:integer;
    e:boolean;
    a:text;
procedure fichier;
  begin
  clrscr;
  writeln('nom du fichier à lire ?');
  readln(h);
  h:=h+'.rom';
  assign(a,h);
  reset(a);
  f:=0;
  repeat
    read(a,m[f]);
    e:=seekeof(a);
    f:=f+1;
  until e=true;
  close(a);
  end;
procedure vitesse;
  begin
  clrscr;
  writeln('vitesse ?');
  readln(v);
  end;
procedure choix;
  begin
  clrscr;
  writeln('adresse de debut ?');
  readln(ad);
  writeln('adresse de fin ?');
  readln(af);
  for g:=ad to af do
    begin
    port[888]:=m[g];
      for j:=0 to v do
      begin
      end;
    end;
  if ad<>af then choix;
  end;
procedure boucle;
  begin
  choix;
  vitesse;
  if v<>0 then boucle;
  end;
begin
  fichier;
  vitesse;
  boucle;
end.
(* COPYRIGHT 1991 Patrick GUEULLE *)
```

Fig. 3.12. – Le logiciel de repérage.

Une fois les parties intéressantes de plusieurs sons ainsi repérées, le petit programme BASIC de la figure 3.13 permet de procéder à leur montage dans un nouveau fichier. Bien entendu, la durée du son ainsi construit ne devra pas excéder la capacité du montage chargé de le reproduire.

```
10 REM ----- MONTAGE.BAS -----
20 CLS:PRINT"Fichier .ROM de destination ?"
30 INPUT FD$:FD$=FD$+".ROM"
40 OPEN FD$ FOR OUTPUT AS#2:K=0
50 CLS:PRINT"Fichier .ROM à ajouter ?"
60 PRINT"(ENTER pour finir)"
70 PRINT"NOM":INPUT F$:IF F$="" THEN 190
80 F$=F$+".ROM"
90 OPEN F$ FOR INPUT AS#1
100 PRINT"DEBUT":INPUT D
110 PRINT"FIN":INPUT F
120 CLS:PRINT"---- MONTAGE EN COURS ----"
130 FOR G=1 TO D-1
140 INPUT#1,N:NEXT G
150 FOR G=D TO F
160 INPUT#1,N:K=K+1
170 PRINT#2,N;:NEXT G
180 CLOSE#1:GOTO 50
190 CLOSE
200 CLS:PRINT FD$;" est disponible"
210 PRINT"(";K;" octets )"
220 BEEP:END
230 REM (c)1991 Patrick GUEULLE
```

Fig. 3.13. – Le logiciel de montage.

Liste des composants du reproducteur de contrôle :

Résistances (5% 1/4 W) :

R1 à R9 : 100 kΩ,
R10 à R16 : 47 kΩ,
R17 : 560 Ω,
R18 : 220 kΩ,
R19 : 330 kΩ,
R20 : 1,5 kΩ,
R21 : 47 kΩ.

Condensateurs :

C1 : 22 µF 16 V radial,
C2 : 10 nF,
C3 : 47 nF.

Transistors :

T1 : 2N 2907.

Divers :

1 embase DIN 5 broches 45° pour circuit imprimé.

UN REPRODUCTEUR DE SON AUTONOME A EPROM

Le schéma de la figure 3.14 n'utilise qu'un petit nombre de composants courants et bon marché autour de la mémoire EPROM 27C64, qu'il n'y a aucun inconvénient à choisir en boîtier plastique OTP dès lors que le son à reproduire est définitivement figé : moyennant l'effacement de son marquage, on peut alors facilement la faire passer pour un circuit intégré spécifique, ce dont les fabricants de kits ne se privent guère !

Un compteur 4040 entraîné par un oscillateur à portes CMOS suffit pour balayer la moitié des 8192 adresses de la mémoire, dont le bus de données attaque un convertisseur numérique-analogique simplifié à l'extrême : un classique réseau de résistances R-2R (ou à peu près !).

S'y ajoutent un amplificateur audio (LM 386 pouvant attaquer directement un haut-parleur, un amplificateur plus puissant ou un transformateur de ligne téléphonique), un régulateur de tension, et une logique de déclenchement qui, grâce à un relais REED, permet au montage de ne strictement rien consommer en veille : il peut donc très bien fonctionner sur une simple pile de 9 V.

Le choix entre l'une ou l'autre des deux moitiés de la mémoire (et donc entre les deux sons possibles) se fait par positionnement d'un petit cavalier reliant la ligne d'adresse A12 à la masse ou au +5 V.

Le déclenchement est prévu par un appui bref sur un bouton-poussoir, mais pourrait tout aussi bien être confié à un automatisme tel que détecteur de présence, barrière infrarouge, séquenceur, ou automate programmable.

Tout le montage tient sur un circuit imprimé simple face dont la figure 3.15 fournit le tracé. L'implantation selon la figure 3.16 fait apparaître quelques straps, qui se sont révélés nécessaires pour échapper à la technique double face : il conviendra de ne pas les oublier !

Si l'on souhaite pouvoir changer de mémoire (et donc de sons) de temps à autre, il sera préférable de faire appel à un support à 28 broches. Dans le cas contraire, rien n'interdit de souder la mémoire comme les autres circuits intégrés.

Le timbre du son et corrélativement sa durée, peuvent être réglés à l'aide d'un potentiomètre ajustable. Il est d'ailleurs souvent intéressant de modifier légèrement le timbre originel, notamment lorsqu'on ne souhaite pas que l'on puisse reconnaître la voix du locuteur. Oui, la fidélité du système serait suffisante pour cela, à condition évidemment de l'utiliser avec soin, et en particulier de l'équiper d'un haut-parleur de bonne qualité (petite enceinte).

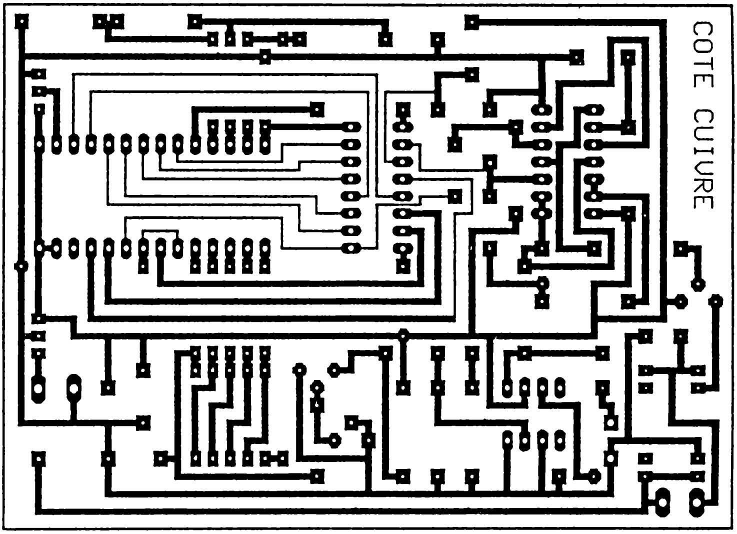


Fig. 3.15. – Circuit imprimé du reproducteur autonome.

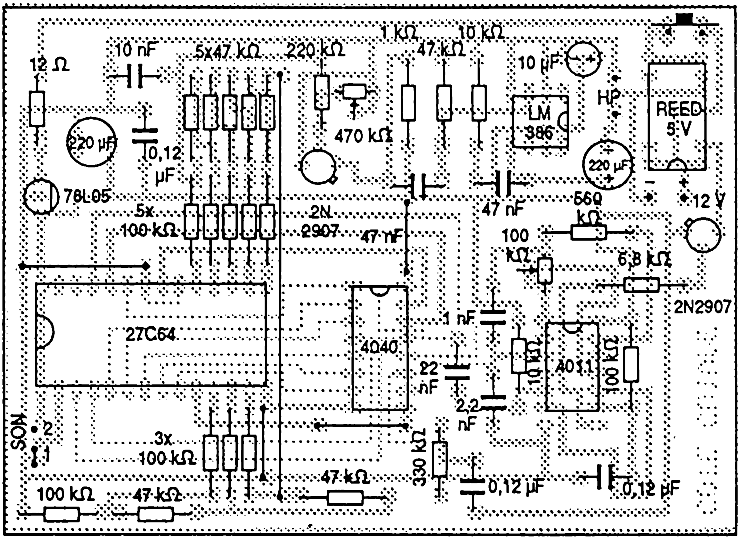


Fig. 3.16. – Plan de câblage du reproducteur autonome.

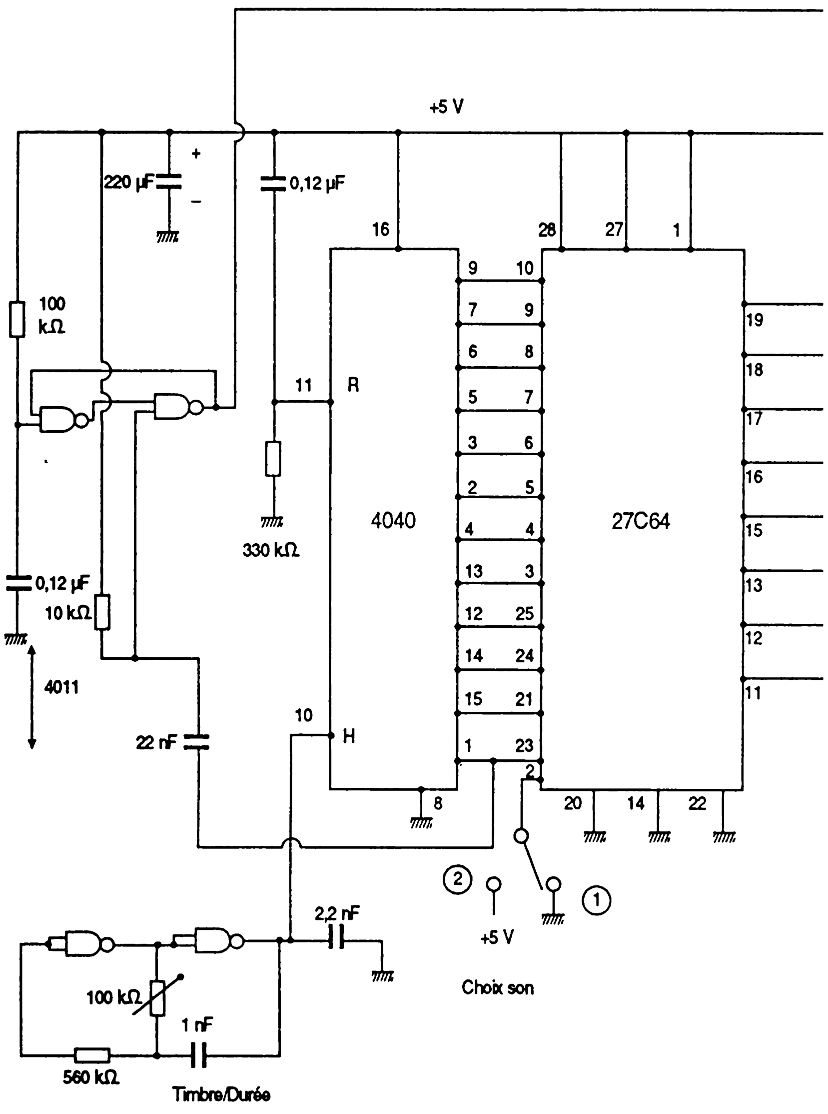
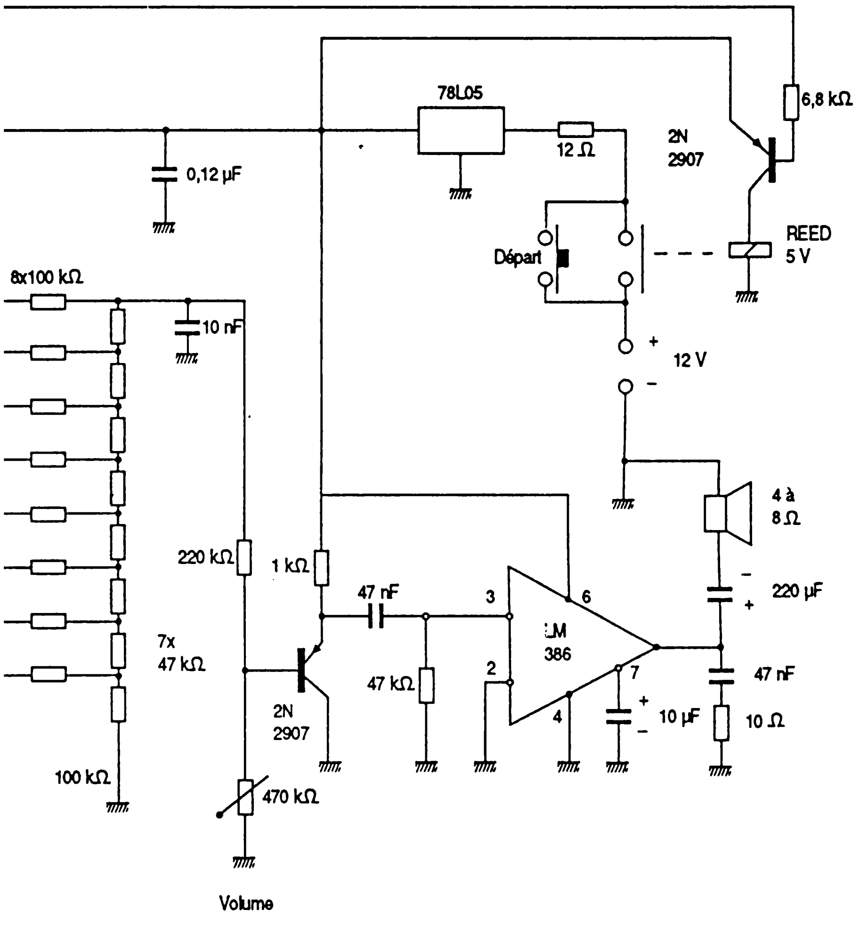


Fig. 3.14. – Schéma du reproducteur autonome.



78L05
0,12 µF
12 Ω
2N
2907
6,8 kΩ
REED
5 V
Départ
8x100 kΩ
10 nF
+
12 V
-
4 à
8 Ω
220 kΩ
1 kΩ
47 nF
3
6
LM
386
-
220 µF
+
7x
47 kΩ
47 kΩ
2
7
47 nF
2N
2907
4
+
10 µF
-
10 Ω
100 kΩ
470 kΩ
Volume

Liste des composants du reproducteur à EPROM :

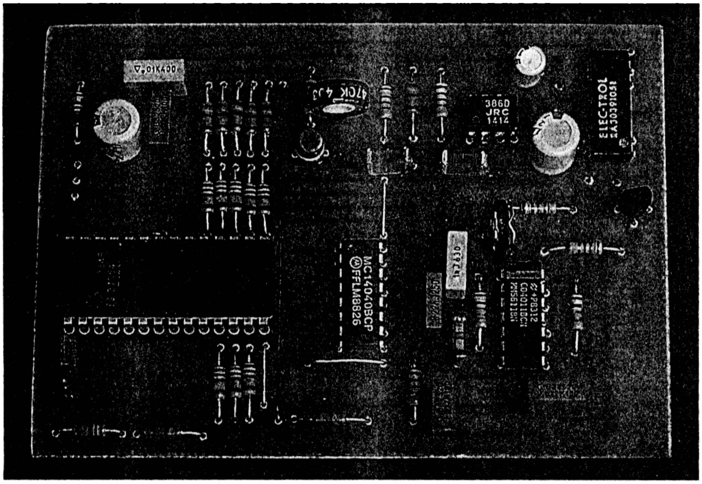
Résistances (5% 1/4 W) :
R1 : potentiomètre ajustable 100 kΩ,
R2 : 560 kΩ,
R3 : 10 kΩ,
R4 : 100 kΩ,
R5 : 330 kΩ,
R6 à R14 : 100 kΩ,
R15 : potentiomètre ajustable 470 kΩ,
R16 : 220 kΩ,
R17 : 1 kΩ,
R18 : 47 kΩ,
R19 : 10 Ω,
R20 : 12 Ω,
R21 : 6,8 kΩ.

Condensateurs :
C1 : 1 nF,
C2 : 2,2 nF,
C3 : 22 nF,
C4 : 0,12 µF,
C5 : 220 µF 10 V radial,
C6 : 0,12 µF,
C7 : 0,12 µF,
C8 : 10 nF,
C9 : 47 nF,
C10 : 10 µF 10 V radial,
C11 : 47 nF,
C12 : 220 µF 10 V radial.

Transistors :
T1 : 2N 2907,
T2 : 2N 2907.

Circuits intégrés :
CI1 : CD 4011B,
CI2 : CD 4040B,
CI3 : 27C64 à programmer,
CI4 : LM 386,
CI5 : 78L05.

Divers :
1 relais REED 5 V DIL,
1 haut-parleur 4 à 8 Ω,
1 bouton-poussoir,
1 cavalier à 2 positions,
1 support tulipe 28 contacts.



UN SYSTÈME DE DÉVELOPPEMENT POUR SYNTHÈSE DE SONS

Le circuit intégré spécialisé UM 5100 (fabriqué par UMC), dont le schéma synoptique est reproduit à la figure 3.17, est ce qu'il est convenu d'appeler un magnétophone statique : associé à une mémoire de capacité suffisante, il est capable d'enregistrer et reproduire avec une qualité honorable, des sons de toutes sortes dont la durée peut approcher la minute.

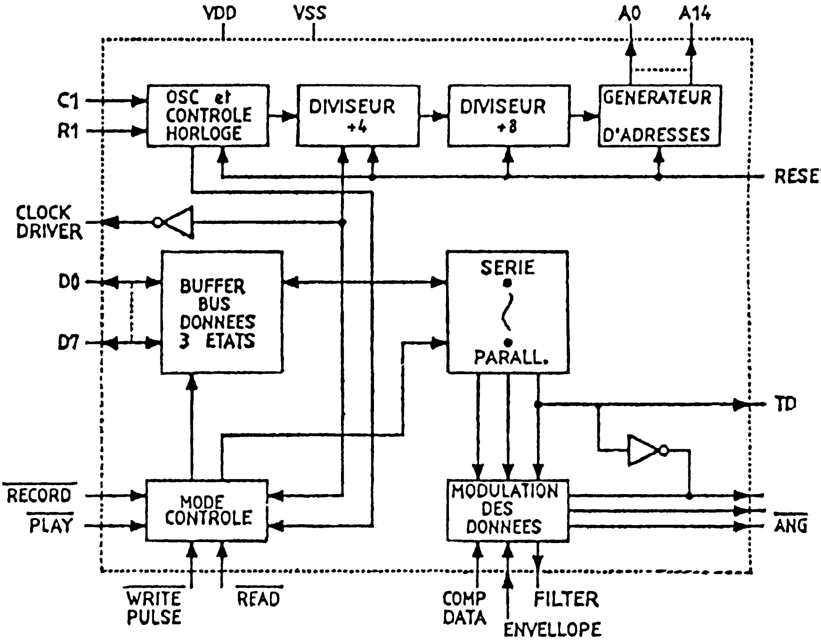


Fig. 3.17. – Synoptique interne de l'UM 5100.

Il contient un modulateur-démodulateur CVSD performant, mais aussi toute la logique nécessaire à l'interfaçage avec une mémoire statique organisée en mots de huit bits. Ce choix peut sembler surprenant du fait que la modulation delta fournit des mots d'un seul bit : la plupart des produits concurrents opèrent en association avec des mémoires dynamiques organisées en mots d'un bit.

L'avantage déterminant de la solution adoptée par UMC est que toutes les mémoires conformes au standard BYTEWIDE se trouvent ainsi être compatibles avec l'UM 5100 : RAM, ROM, EPROM, et même ZEROPOWER : il en résulte une extrême souplesse d'emploi, rendant inutiles les coûteux systèmes de développement imposés par la plupart des marques pour programmer des sons dans leurs composants !

Avantage supplémentaire apparaissant à la figure 3.18, l'UM 5100 est offert en boîtier DIP à 40 broches, contrairement à beaucoup de ses concurrents disponibles seulement en CMS (pour montage en surface) : il se prête donc idéalement à la réalisation de pièces uniques ou de très petites séries de *moutons à cinq pattes* répondant à des besoins ponctuels.

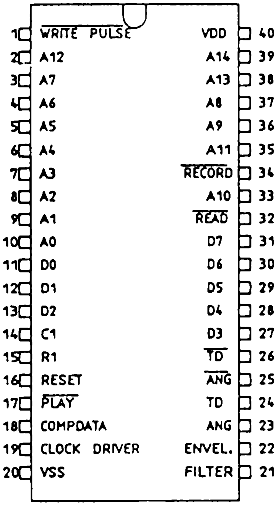


Fig. 3.18. – Brochage de l'UM 5100.

Le principal inconvénient de ce merveilleux composant est qu'il ne possède pas d'équivalent strict ou seconde source : la faisabilité des trois montages qui vont être décrits est donc liée à la disponibilité de l'UM 5100 dans le commerce.

A l'heure où nous écrivons ces lignes, celle-ci ne pose aucun problème, mais un jour viendra forcément où sa fabrication sera abandonnée au profit de solutions encore meilleures. Il faudra alors réétudier tous les circuits d'application : telle est la rançon de l'évolution si rapide de ces techniques de pointe !

L'affectation des broches de l'UM 5100 est la suivante :

VDD	Alimentation positive 3 à 6 V
VSS	Masse (0 V)
A0 à A14	Bus d'adresse pour la mémoire
D0 à D7	Bus de données pour la mémoire
/RECORD	Commande d'enregistrement (fugitive)
/WRITE PULSE	Impulsion d'écriture en mémoire
/READ	Impulsion de lecture en mémoire
/PLAY	Commande de lecture (fugitive)
RESET	Remise à zéro et mise en attente
ANG et /ANG	Sorties analogiques
FILTER	Sortie pour filtre d'enveloppe
ENVELOPE	Entrée pour filtre d'enveloppe
TD et /TD	Sorties auxiliaires audio
COMPDATA	Reçoit sortie du comparateur externe
C et R	Oscillateur RC pour l'horloge
CLOCK DRIVER	Signal d'horloge bufferisé

Cela étant précisé, la figure 3.19 montre le principe d'un magnétophone bâti autour de l'UM 5100 associé à quelques amplificateurs opérationnels (préamplificateur, filtres, comparateur) et à une mémoire : RAM pour l'enregistrement-lecture, ROM ou EPROM pour la lecture seule.

La prise de son est confiée à un micro à électret muni d'un préamplificateur précédant le comparateur, tous deux réalisés à partir de la moitié d'un LM 324 tirant son alimentation négative de la sortie d'horloge bufferisée, grâce à un doubleur de tension à diodes.

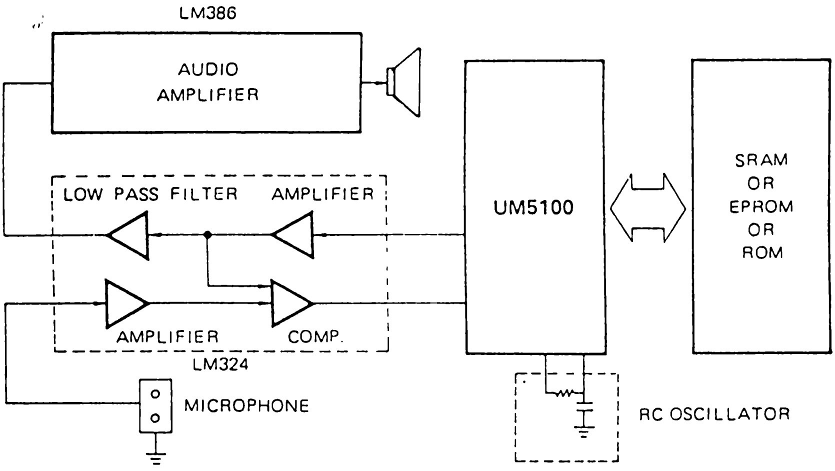


Fig. 3.19. – Principe d'un magnétophone statique à UM 5100.

Les deux amplificateurs opérationnels restants servent à réaliser le sommateur combinant les diverses sorties audio du démodulateur CVSD (hautes et basses fréquences étant séparées pour une meilleure qualité de son), et le filtre actif de limitation de la bande passante à l'intervalle utile.

Un LM 386 se charge enfin de piloter le haut-parleur : maintenu bloqué tant que le signal /READ est au niveau haut, il ne délivre de son que pendant les seules périodes de lecture effective.

Précisons bien qu'une qualité de son convenable ne peut guère être obtenue que sur un haut-parleur raisonnablement performant : un simple HP miniature de 5 cm dépourvu de coffret reproduira forcément un son décevant.

L'utilisation est extrêmement simple : après un bref appui sur le bouton d'enregistrement, parler dans le micro et constater les variations d'éclairement de la diode LED de contrôle de modulation, qui s'éteint lorsque la mémoire est pleine.

Appuyer alors brièvement sur le poussoir de lecture pour écouter le son enregistré. Le cas échéant, il est possible de faire varier, en continu, le timbre du son reproduit en jouant sur la fréquence d'horloge : rien de tel pour rendre une voix méconnaissable !

Le bus d'adresses à quinze lignes permet l'emploi de mémoires de capacité pouvant aller jusqu'à 32 K-octets, soit une trentaine de secondes de son de qualité téléphonique (8 K-bits par seconde). Certaines applications se contentent cependant de nettement moins, à raison d'environ 1 K-octets par seconde de parole (mais on peut préférer améliorer la qualité du son en accroissant ce débit et donc la taille mémoire...)

L'UM 5100 étant supposé au repos (application d'une impulsion positive sur sa broche RESET lors de la mise sous tension), l'application d'une impulsion négative sur /RECORD ou sur /PLAY déclenche le démarrage d'un cycle d'enregistrement ou de lecture : les adresses de la mémoire sont parcourues dans l'ordre croissant, avec bouclage pur et simple lorsque la capacité maximum est atteinte.

A moins de couper l'alimentation (et donc de perdre le contenu de la RAM s'il ne s'agit pas d'un modèle à pile incorporée), le seul moyen d'arrêter un cycle de lecture consiste à appliquer une impulsion de RESET. On peut faire de même pour interrompre un cycle d'enregistrement, mais il est aussi possible de passer à tout moment en mode lecture, par une simple impulsion négative appliquée à la broche /READ.

On peut ainsi songer à pratiquer un enregistrement permanent (en boucle), et à l'interrompre dès que l'on souhaite que le système répète ce qui vient d'être dit, plusieurs fois si nécessaire. Le tout bien sûr sans bande magnétique ni tête, dont l'usure serait rapide en cas d'usage permanent.

La plupart du temps, on détecte la fin d'un cycle par le biais du front descendant de la dernière ligne d'adresse desservant la mémoire utilisée, ou du front montant de la ligne suivant celle-ci : la première méthode permet de passer en mode lecture dès la fin d'un enregistrement grâce à un simple condensateur placé entre /PLAY et la dernière ligne d'adresse utilisée, tandis que la seconde permet un arrêt pur et simple en reliant la première ligne d'adresse inutilisée à RESET.

Il est même possible de combiner les deux procédés : après un cycle complet d'enregistrement, un cycle de lecture (contrôle) est exécuté, puis tout s'arrête. C'est classique dans le domaine des répondeurs téléphoniques.

Seulement, réserver une ligne d'adresse pour la commande du RESET divise par deux la capacité d'adressage mémoire de l'UM 5100, qui passe alors de 32 K-octets à 16 K-octets, soit 128 K-bits. Cela ne correspond guère qu'à 16 secondes de son de bonne qualité, échantillonné à 8 K-bits par seconde...

Qu'à cela ne tienne ! Rien n'est plus simple que de monter un compteur d'adresses additionnel en cascade avec celui de l'UM 5100 : compteur binaire de type 4040 pour adresser des boîtiers de plus de 16 ou 32 K-

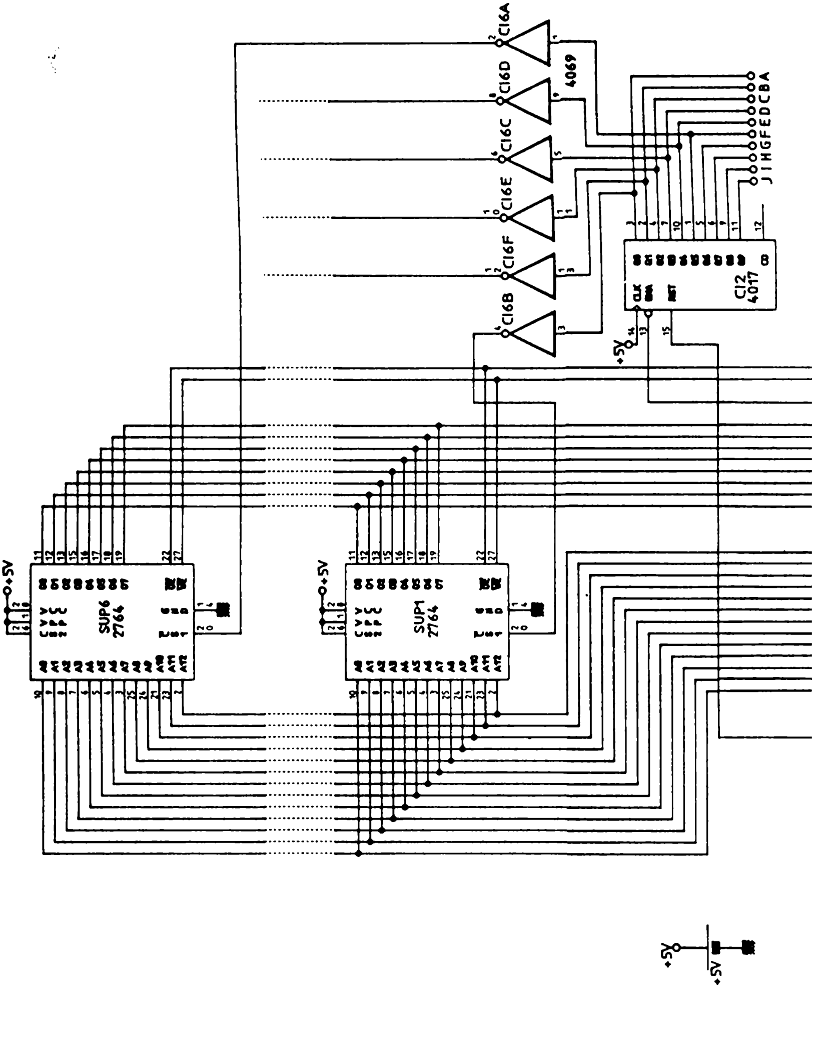
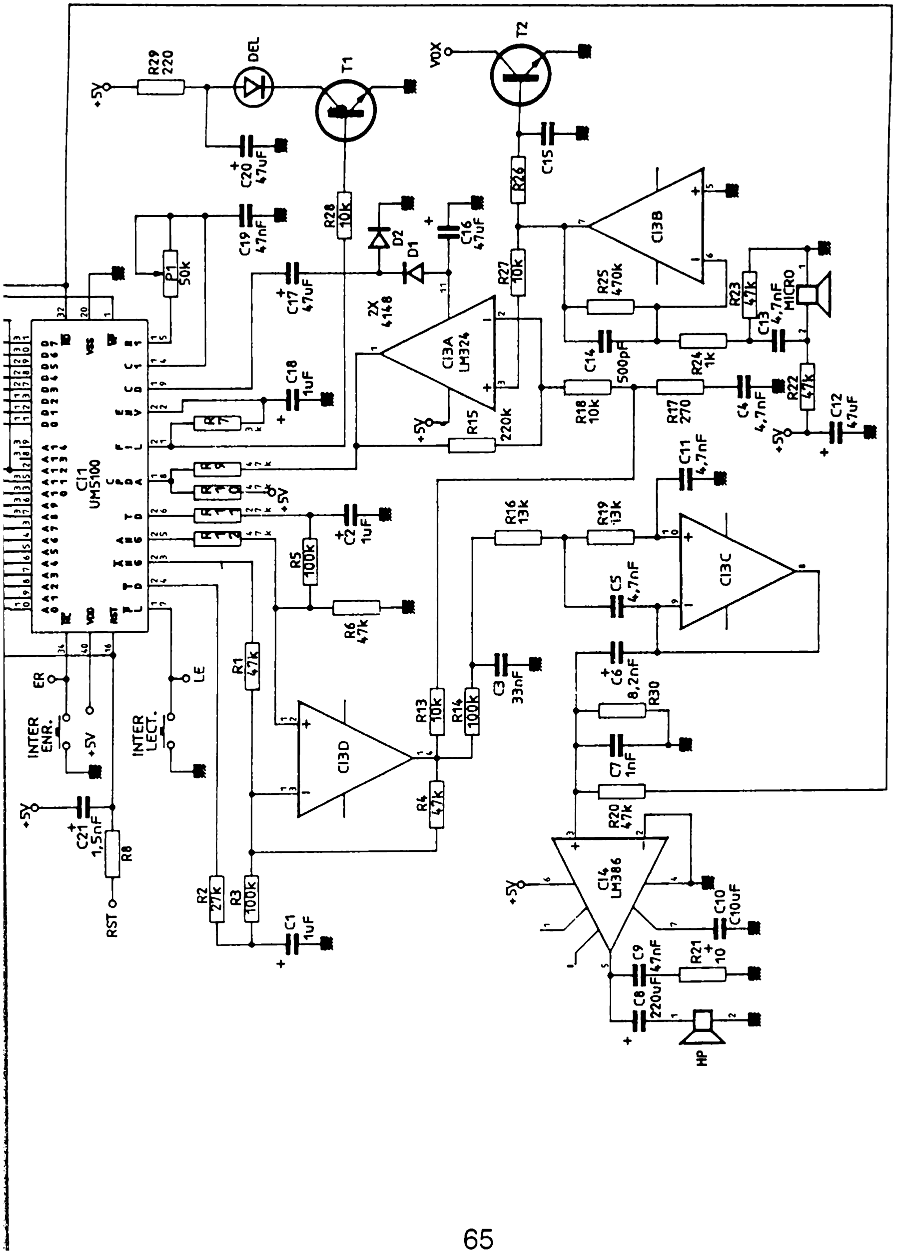


Fig. 3.20. – Schéma de la carte de développement pour UM 5100.


CI1
UM5100
CI3A
LM324
CI3B
CI3C
CI3D
CI4
LM386
T1
T2
DEL
MICRO
HP
VOX
+5V
INTER
ENR.
INTER
LECT.
ER
LE
RST
R29
220
C20
47uF
C19
47nF
P1
50k
R28
10k
D2
D1
2X
4148
C16
47uF
C17
47uF
C18
1uF
C15
R26
R27
10k
R25
470k
R23
47k
C13
4,7nF
C14
500pF
R24
1k
R18
10k
R17
270
C4
4,7nF
R22
47k
C12
47uF
R15
220k
C11
4,7nF
R16
13k
R19
13k
C5
4,7nF
C2
1uF
R5
100k
R6
47k
C6
8,2nF
R30
C7
1nF
R20
47k
C3
33nF
R13
10k
R14
100k
R4
47k
R1
47k
R2
27k
R3
100k
C1
1uF
C21
1,5nF
R8
C10
C10uF
C9
47nF
R21
10
C8
220uF

octets et comportant donc des lignes d'adresse supplémentaires, ou compteur décimal genre 4017 pour commuter plusieurs boîtiers de capacité normale.

On obtiendra par exemple le même résultat en utilisant une seule EPROM 27256, ou bien quatre 2764. Au niveau de la prise de son, il sera par contre plus économique d'utiliser quatre fois de suite une mémoire RAM non volatile de 8 K-octets (48Z08 de SGS THOMSON) que son équivalent de 32 K-octets qui coûte tout de même beaucoup plus cher.

Il suffira de préparer au préalable un enregistrement sur bande de bonne qualité, et de bien prendre ses repères lors des quatre phases successives de transfert en mémoire.

Le schéma de notre carte de développement, reproduit à la figure 3.20, tient le plus grand compte de cette analyse : il associe l'UM 5100 à un compteur 4017 chargé de commuter six banques de mémoire de 64 K-bits chacune (EPROM 2764 ou 27C64, RAM CMOS 6264, ou ZEROPOWER 48Z08 non volatile et recopiable dans une EPROM).

Toutes les mémoires sont câblées en parallèle, à l'exception de leurs lignes de sélection, activées tour à tour par le 4017 relayé par les six inverseurs d'un 4069.

Une simple connexion entre le point RST (RESET) et l'un des points B à G suffit pour fixer le nombre de boîtiers enregistrés ou lus : le tableau de la figure 3.21 récapitule les capacités mémoire ainsi obtenues, et les durées correspondantes pour trois fréquences d'échantillonnage menant à différents niveaux de qualité du son (bon, moyen, et médiocre mais intelligible).

RST relié à	Nombre de boîtiers mémoire	Capacité (k-bits)	Durée à 8 K-bits/s	Durée à 4 K-bits/s	Durée à 16 K-bits/s
B	1	64	8 s	16 s	4 s
C	2	128	16 s	32 s	8 s
D	3	192	24 s	48 s	12 s
E	4	256	32 s	64 s	16 s
F	5	320	40 s	80 s	20 s
G	6	384	48 s	96 s	24 s

Fig. 3.21. – Principales configurations mémoire possibles.

Bien entendu, il est possible de monter tous les boîtiers et d'en utiliser tout ou partie selon la situation du moment, d'en insérer moins de six, et de panacher EPROM, RAM, et ZEROPOWER sur un même montage : la configuration des supports a été prévue en conséquence.

Ce schéma ne se sert pas des lignes A13 et A14 de l'UM 5100, ce qui permettrait facilement de doubler ou quadrupler la capacité de la carte en employant des boîtiers de 128 ou 256 K-octets (il suffirait d'ajouter les lignes nécessaires au bus d'adresse, et de commander le 4017 non plus par A12, mais par A13 ou A14).

Dans l'état actuel des choses, une telle extension resterait cependant trop onéreuse pour une majorité d'applications. Retenons tout de même qu'elle permettrait d'atteindre six minutes de son de qualité téléphonique avec six 27256 : de quoi dire bien des choses, par exemple dans des applications muséographiques.

Le reste du montage est conforme aux schémas d'application conseillés par UMC, et comporte un détecteur de son permettant un déclenchement automatique de l'enregistrement ou de la lecture (connecter le point VOX à ER ou à LE respectivement).

La relative complexité des interconnexions a rendu nécessaire le recours à la technique double face, sans pour autant exiger de trous métallisés. Quelques vias supposent cependant un soudage de certains composants sur les deux faces. Les figures 3.22 et 3.23 fournissent les tracés des deux cuivres, tandis que la figure 3.24 reproduit le plan de câblage.

L'implantation exige beaucoup de soin : il sera d'ailleurs prudent de contrôler les continuités et l'absence de court-circuits au niveau des bus mémoire.

On aura tout intérêt à commencer les essais avec un seul boîtier mémoire (par exemple une 6264), placé dans le support N°1. Le point RST étant donc relié au point B, on doit obtenir l'équivalent d'un petit magnétophone capable d'enregistrer et de relire une dizaine de secondes de son : une variante en somme du synthétiseur de son déjà décrit dans notre ouvrage *ALARME ET SURVEILLANCE A DISTANCE.*

La durée exacte d'enregistrement (et donc la qualité du son) se règle très facilement avec un petit potentiomètre ajustable fixant la fréquence d'horloge de l'UM 5100.

Sauf problème qu'il faudrait impérativement résoudre avant d'aller plus loin, on pourra alors insérer un second boîtier dans le support N°2 (et donc déplacer RST de B en C), et ainsi de suite jusqu'à concurrence de six (RST devant alors rejoindre le point G).

Bien évidemment, ces connexions du point RST pourraient fort bien être confiées à une logique extérieure à la carte, pouvant même gérer aussi le démarrage des cycles d'enregistrement et de lecture. A la limite, l'emploi d'un microcontrôleur ou d'un ordinateur ne serait pas ridicule à ce niveau, mais nous y reviendrons...

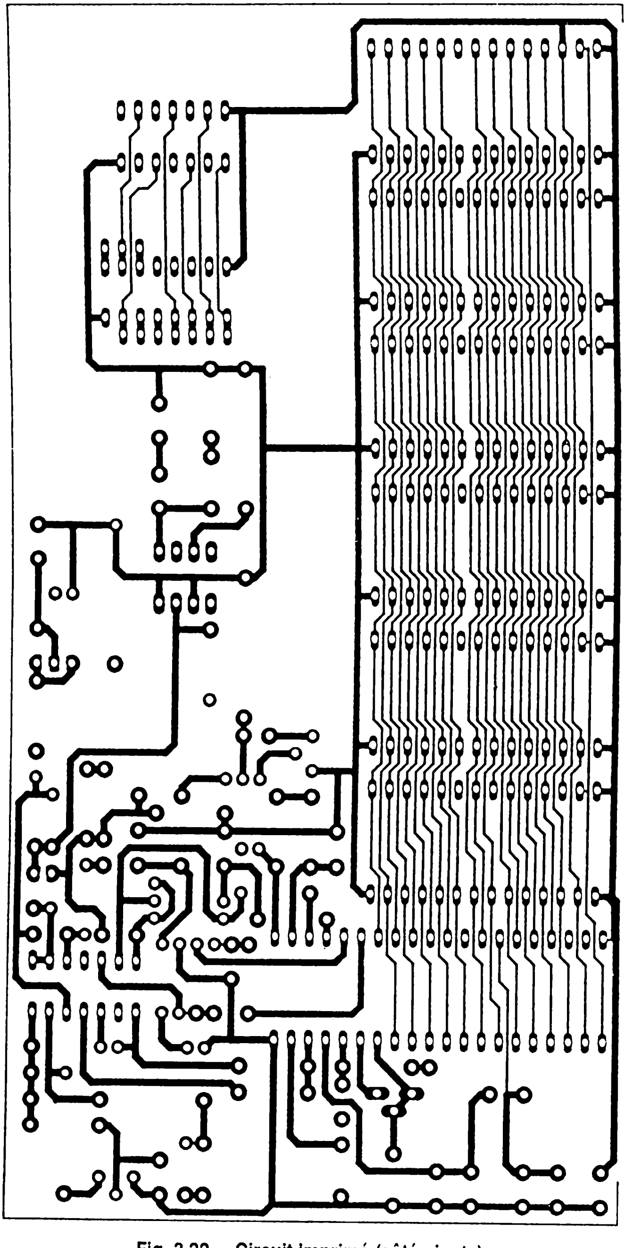


Fig. 3.22. – Circuit imprimé (côté picots).

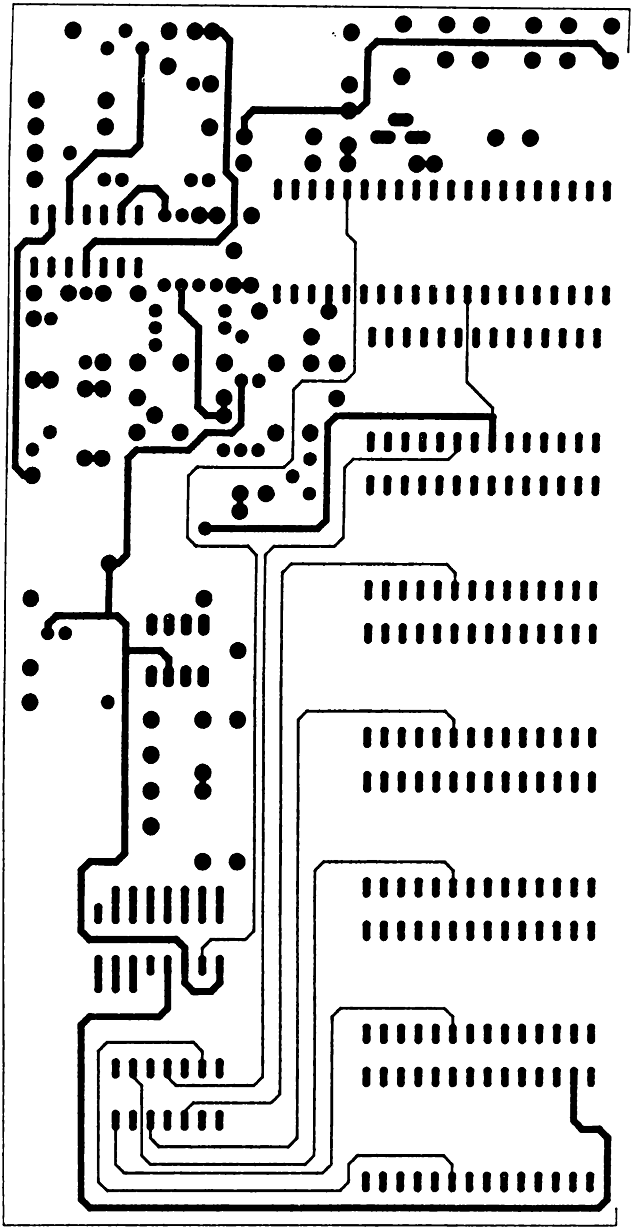


Fig. 3.23. – Circuit imprimé (côté composants).

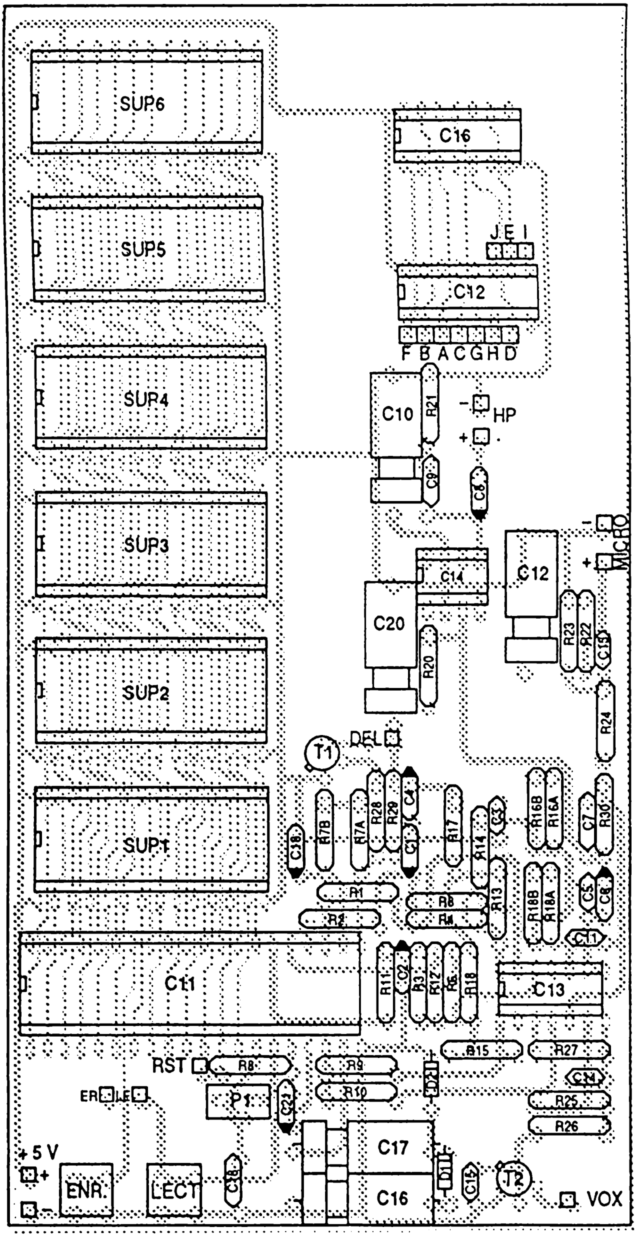


Fig. 3.24. – Plan de câblage de la carte de développement.

De même, rien n'empêche de supprimer le 4017 et le 4069 qui lui est associé : A13 de l'UM 5100 rejoignant directement RST, une logique indépendante pourrait alors se charger de choisir parmi les six disponibles, l'unique boîtier qui serait lu ou enregistré !

Avec un peu plus d'imagination, on peut aussi songer à appeler les différents boîtiers dans un ordre variable en fonction des circonstances, ou même avec des répétitions : de quoi composer des messages aux significations fort différentes. Dans tous les cas, on gardera présentes à l'esprit les possibilités des trois types de mémoires que l'on peut monter, ou même panacher, sur cette carte :

- Les RAM (6264) peuvent être enregistrées, lues, et surchargées à volonté, mais perdent leur contenu dès la coupure de l'alimentation. Une RAM non enregistrée produit un son aléatoire, en général un puissant grésillement.

- Les ZEROPOWER (48Z08) se comportent comme les RAM mais conservent leur contenu en l'absence d'alimentation, même si on les débroche. Elles risquent évidemment d'être effacées par mégarde à l'occasion d'une opération d'enregistrement malencontreuse. On peut par contre très facilement les copier dans des EPROM à l'aide d'un programmateur classique.

- Les EPROM (2764 ou 27C64) ne peuvent être programmées que dans un programmateur spécial, utilisant comme modèle une 48Z08 enregistrée sur notre carte, et qui pourra ensuite être réutilisée pour un nombre illimité d'autres prises de son. Les EPROM existent en boîtier céramique à fenêtre de quartz (effaçables aux ultraviolets) ou en boîtier plastique OTP ineffaçables mais beaucoup plus économiques.

Une EPROM vierge (tous les bits à 1) produit un puissant sifflement, tandis qu'un support vide se traduit généralement par le silence et par l'allumage permanent de la LED servant de modulomètre.

A chacun de choisir en fonction de ses applications propres !

Liste des composants de la carte de développement :

Résistances (5% 1/4 W) :
P1 et P2 : potentiomètres ajustables 47 ou 50 kΩ,
R1, R4, R6, R8, R9, R10, R12, R20, R22, R23, R25 : 47 kΩ,
R2 et R11 : 27 kΩ,
R3, R5, R14 :100 kΩ,
R7 : 3 kΩ (ou 3,3 kΩ),
R13, R18, R26, R27, R28 : 10 kΩ,
R15 : 220 kΩ,
R16 et R19 : 13 kΩ (ou 12 kΩ),

R17 : 270 kΩ,
R21 : 10 Ω,
R24 : 1 kΩ,
R29 : 220 Ω,
R30 : 2,2 kΩ.

Condensateurs :
C1, C2, C18 : 1 µF 16 V,
C3 : 33 nF,
C4, C5, C11, C13 : 4,7 nF,
C6 : 8,2 nF,
C7 : 1 nF,
C8 : 220 µF 16 V,
C9 et C19 : 47 nF,
C10 : 10 µF 16 V,
C12, C16, C17, C20 : 47 µF 16 V,
C21 : 1,5 nF.

Transistors :
T1 et T2 : 2N 2222 ou BC 107.

Circuits intégrés :
CI1 : UM 5100,
CI2 : CD 4017B,
CI3 : LM 324,
CI4 : LM 386,
CI5 : voir texte,
CI6 : CD 4069B.

Autres semiconducteurs :
D1 et D2 : 1N 4148.

Divers :
2 touches D10 Jeanrenaud,
1 HP 8 à 16 Ω,
1 micro électret 2 fils.

UN REPRODUCTEUR DE SON A UM 5100

Les applications pratiques ne manquent pas, nous l'avons vu plus avant dans ce chapitre, pour une carte autonome capable de reproduire un son préalablement digitalisé et brûlé dans une EPROM.

Grâce à l'UM 5100, nous allons pouvoir passer à des sons durant près d'une minute : déjà de véritables messages, par exemple des explications ou des mises en garde qui pourront être répétées aussi souvent que nécessaire en toute fiabilité.

Susceptible d'être déclenché par un contact ou un détecteur (notamment à infrarouge passif), ce montage trouvera facilement sa place dans les systèmes d'alarme, les animations commerciales, les expositions, ou la signalisation.

Un autre avantage de cette technique est qu'il n'y a plus besoin d'ordinateur pour la prise de son, ni même à la limite de magnétophone : la digitalisation du son peut être effectuée directement sur la carte de développement que nous venons de décrire.

Le schéma de la figure 3.25 exploite au maximum la capacité d'adressage mémoire de l'UM 5100, qui est de 32 K-octets (15 lignes d'adresse), et permet donc l'usage direct d'une EPROM 27256.

Nous avons vu qu'il serait éventuellement possible d'aller plus loin en cascadant un compteur d'adresses supplémentaire, mais là n'est pas notre propos car cela compliquerait sensiblement le montage tout en coûtant nettement plus cher en mémoire. La configuration préconisée par UMC, fabricant de l'UM 5100, a été conservée à quelques détails près :

Tout d'abord, nous avons décidé de prendre l'alimentation de l'amplificateur de puissance avant le régulateur 5 V pour augmenter la puissance disponible.

Ensuite, nous avons arrondi les valeurs des résistances du filtre passe-bas aux valeurs normalisées à 10% les plus proches. Comme on sait que les caractéristiques des filtres actifs sont très sensibles aux valeurs des résistances, nous avons effectué une simulation SPICE pour nous assurer que l'approximation était légitime.

Le circuit de déclenchement est pour sa part un peu plus complexe que celui d'origine : comme nous avions besoin d'une porte NAND pour commander le reset de l'UM 5100 à partir du front arrière du signal d'adresse A14 et lors de la mise sous tension, autant employer les trois portes restantes pour permettre le déclenchement du montage par des impulsions aussi bien positives que négatives.

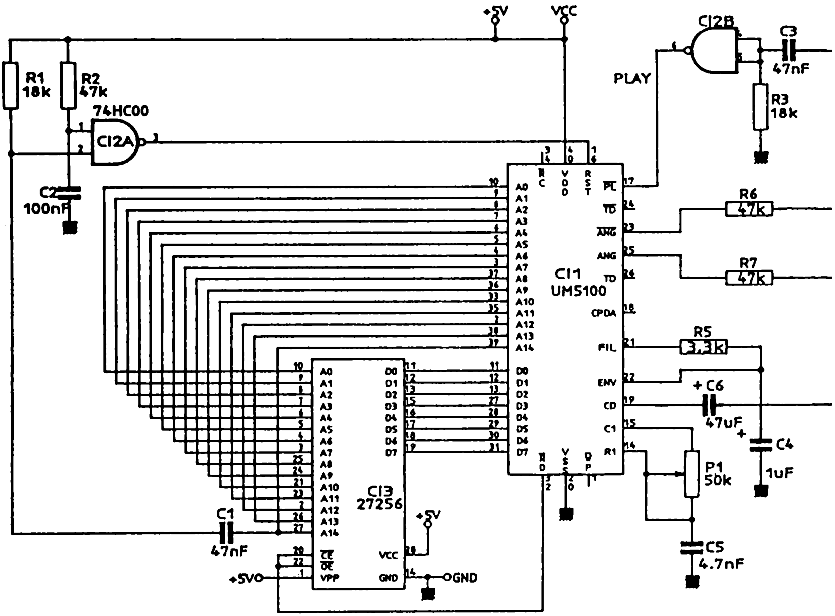
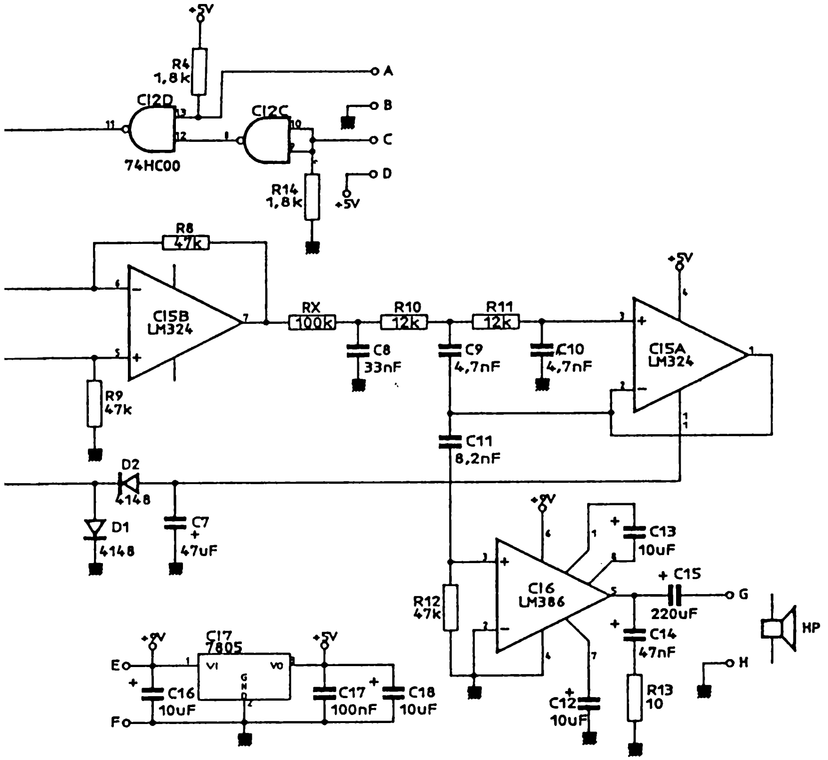


Fig. 3.25. – Schéma du reproducteur de son à UM 5100.


+5V
R4
1,8k
A
CI2D
CI2C
B
C
74HC00
D
R14
1,8k
+5V
R8
47k
CI5B
LM324
RX
100k
R10
12k
R11
12k
+5V
C8
33nF
C9
4,7nF
C10
4,7nF
CI5A
LM324
R9
47k
C11
8,2nF
D2
4148
D1
4148
C7
47uF
+5V
C13
10uF
+ C15
CI6
LM386
G
R12
47k
220uF
HP
C14
47nF
+5V
CI7
7805
+5V
E
VI
VO
GND
C16
10uF
C17
100nF
C18
10uF
F
C12
10uF
R13
10
H

Le reproducteur de son pourra donc être piloté par pratiquement n'importe quoi : de simples contacts, bien sûr, mais aussi des détecteurs à infrarouge passif genre MS 02 (SELECTRONIC), ou tout équipement muni de sorties collecteur ouvert ou TTL (par exemple un séquenceur ou un petit automate programmable tels que ceux décrits au chapitre 2).

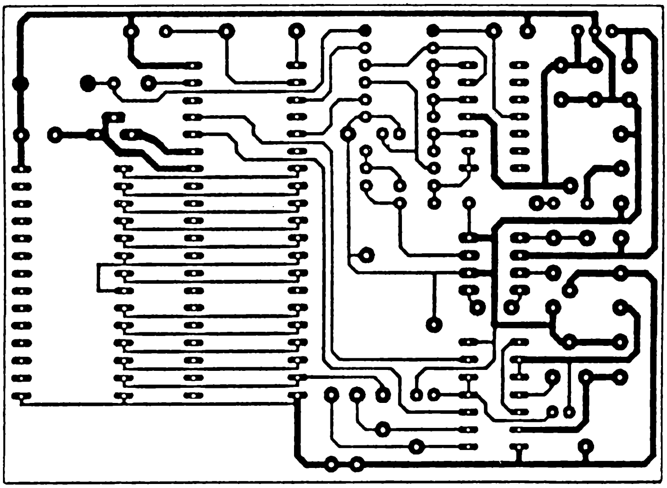


Fig. 3.26. – Circuit imprimé (côté picots).

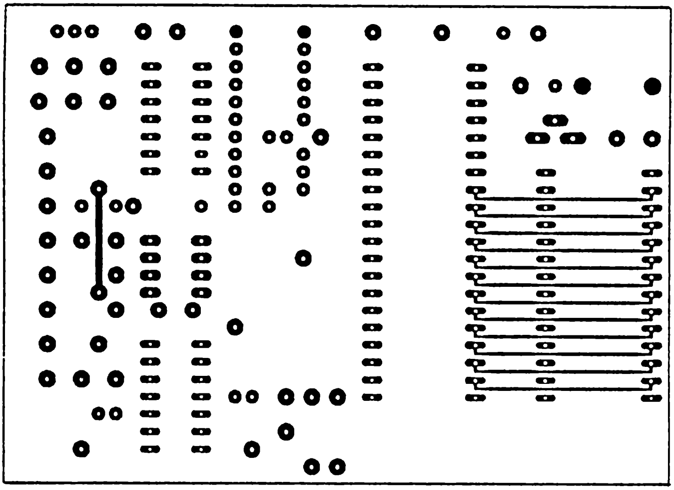


Fig. 3.27. – Circuit imprimé (côté composants).

La construction du montage fait appel à un circuit imprimé double face, dont la figure 3.26 reproduit le tracé côté picots, tandis que la figure 3.27 représente le côté composants.

L'implantation selon la figure 3.28 suppose quelques soudures côté composants, raison pour laquelle il est à recommander d'utiliser des barrettes sécables tulipe en lieu et place des supports à 28 et 40 broches. Huit pastilles au pas de 5,08 mm sont prévues pour le raccordement de l'alimentation (9 à 12 V), du haut-parleur (4 à 50 Ω), et du circuit déclencheur.

L'unique réglage prévu est celui de la vitesse de lecture de la mémoire, qui devra être ajusté en fonction de celui utilisé lors de la prise de son. Rappelons que ce réglage fixe à la fois la durée du son (pour une capacité mémoire donnée), sa qualité, et son timbre dans la mesure où les réglages seraient légèrement différents àl'enregistrement et à la lecture (cela permet par exemple de déguiser efficacement une voix).

La puissance de sortie disponible dépend de l'impédance du haut-parleur et de la tension d'alimentation, et peut être considérée comme suffisante pour la plupart des applications si on fait usage d'une petite enceinte de qualité raisonnable. Pour certaines utilisations, le niveau pourrait même sembler excessif : il suffira alors de supprimer C13 pour diminuer le gain du LM 386.

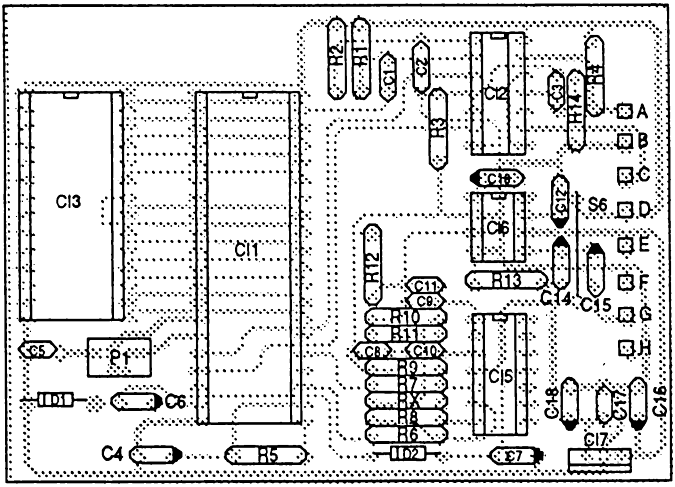
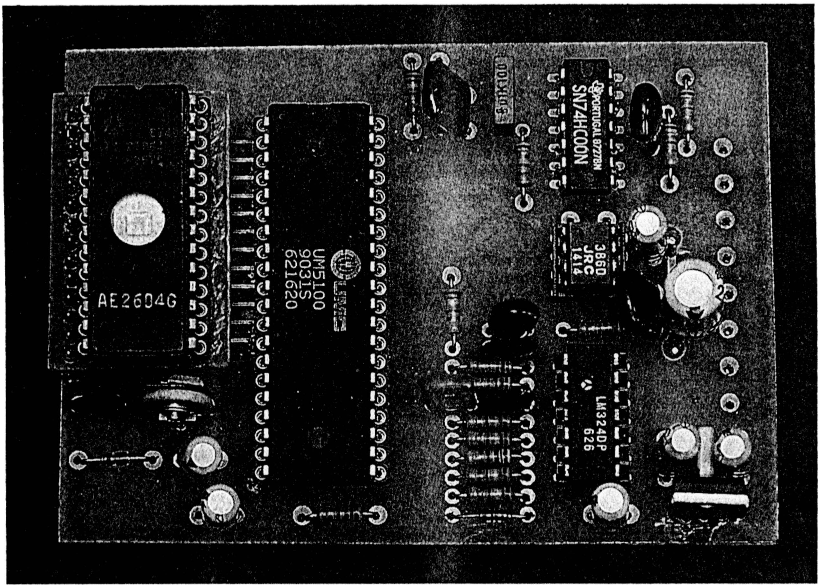


Fig. 3.28. – Plan de câblage du reproducteur de son.



Ce montage étant uniquement lecteur, il faut évidemment réaliser la prise de son au moyen d'un autre équipement, l'idéal étant la carte de développement qui vient d'être décrite.

La digitalisation doit se faire dans une mémoire RAM CMOS à pile lithium incorporée, que l'on recopiera ensuite dans l'EPROM à l'aide d'un programmateur ordinaire.

Notre carte de développement pouvant accueillir jusqu'à six boîtiers de 8 K-octets, rien n'est plus simple que de l'équiper de quatre MK 48Z08 (SGS-THOMSON) ou BQ 4010 (BENCHMARQ).

Mais on peut aussi utiliser, moyennant une modification très simple de la carte (amener les lignes A13 et A14 de l'UM 5100 sur le support de mémoire), une seule BQ 4011. Cette mémoire non volatile de 32 K-octets produite par BENCHMARQ et disponible auprès de NEWTEK correspond à la 27256 qu'utilise notre montage et facilite donc le transfert dans l'EPROM.

Elle possède la particularité d'être munie d'un dispositif qui ne met en service sa pile incorporée qu'à partir de la première mise sous tension de la mémoire : on est ainsi assuré de disposer d'une autonomie maximale (une dizaine d'années), même après un stockage de longue durée du composant neuf.

Comme toutes les RAM statiques de 32 K-octets, la BQ 4011 présente un brochage légèrement différent (figure 3.29) de celui de la 27256 (figure 3.30). On ne peut donc pas monter directement la BQ 4011 contenant le son sur notre reproducteur. Comme un tel contrôle est tout de même utile avant de brûler une EPROM (surtout s'il s'agit d'une OTP ineffaçable), nous avons prévu un adaptateur dont la figure 3.31 donne le schéma.

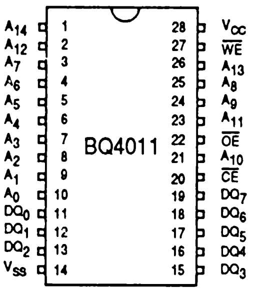


Fig. 3.29. – Brochage de la mémoire non volatile BQ 4011.

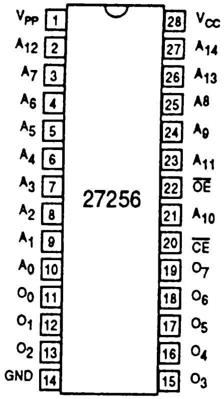


Fig. 3.30. – Brochage de l'EPROM 27256.

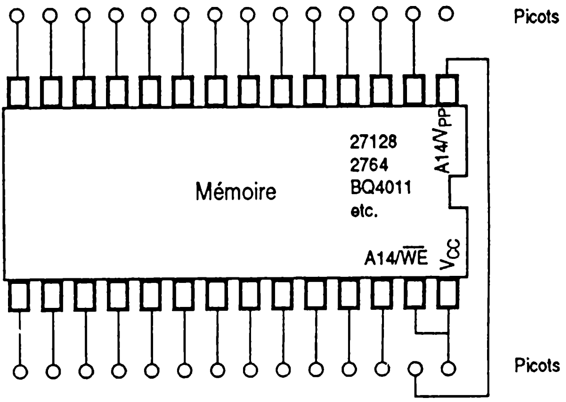


Fig. 3.31. – Schéma de l'adaptateur de mémoires.

On peut le réaliser sur un petit morceau de circuit imprimé dont la figure 3.32 fournit le tracé, et la figure 3.33 le plan de câblage.

La face cuivrée de cette petite carte sera dirigée vers le haut : deux barrettes sécables mâles de 14 picots chacune seront donc soudées normalement, tandis que deux barrettes à 14 tulipes (ou un support à 28 contacts) seront soudées côté cuivre.

Bien que conçu uniquement pour s'intercaler entre le reproducteur de son et une BQ 4011, cet adaptateur peut aussi servir à écouter des 27128 ou des 2764 qui seront bien sûr lues respectivement deux fois et quatre fois à la file. Cela peut même être utile pour des applications nécessitant une certaine insistance !

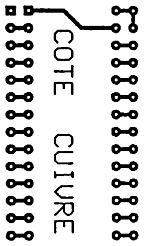


Fig. 3.32. – Circuit imprimé de l'adaptateur de mémoires.

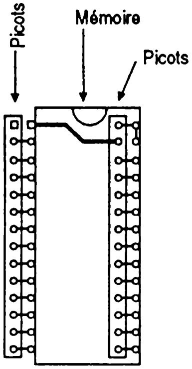


Fig. 3.33. – Plan de câblage de l'adaptateur de mémoires.

La figure 3.34 montre comment brancher un contact de déclenchement sur ce montage, et ausi comment lui associer un détecteur à infrarouge passif MS 02.

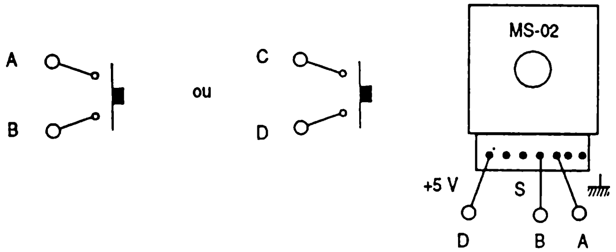


Fig. 3.34. – Branchement d'un détecteur MS-02.

Ce détecteur est alimenté par le régulateur 5 V du montage, et n'a donc pas besoin d'une alimentation séparée. Même sans lentille de Fresnel, sa portée suffit pour déclencher l'émission du son enregistré lorsque l'on passe devant le système. Un bon endroit pour placer le détecteur pourrait donc être, par exemple, le boîtier du haut-parleur dans lequel rien n'interdit d'ailleurs de loger également la carte principale.

On peut ainsi réaliser un ensemble compact, qu'il suffira de compléter par un petit bloc secteur prise de courant délivrant entre 9 et 12 V.

Liste des composants du reproducteur de son :

Résistances (5% 1/4 W) :
R1 et R3 : 18 kΩ,
R2, R6, R7, R8, R9, R12 : 47 kΩ,
R4 et R14 : 1,8 kΩ,
R5 : 3,3 kΩ,
R10 et R11 : 12 kΩ,
R13 : 10 Ω,
Rx : 100 kΩ,
P1 : potentiomètre ajustable 47 ou 50 kΩ.

Condensateurs :
C1, C3, C14 : 47 nF,
C2 et C17 : 0,1 µF,
C4 : 1 µF 16 V,
C5, C9, C10 : 4,7 nF,
C6 et C7 : 47 µF 16 V,
C8 : 33 nF,
C11 : 8,2 nF,
C12, C13, C16, C18 : 10 µF 16 V,
C15 : 220 µF 16 V.

Circuits intégrés :
CI1 : UM 5100,
CI2 : 74HC00,
CI3 : 27C256 ou 27256 à programmer (variantes : voir texte)
CI4 : 7805 (régulateur positif 5 V),
CI5 : LM 324,
CI6 : LM 386.

Autres semiconducteurs :
D1 et D2 : 1N 4148.

UN SYNTHÉTISEUR VOCAL A UM 5100

L'UM 5100 n'est pas à proprement parler un synthétiseur vocal, mais plutôt un magnétophone statique.

Sa vocation est d'enregistrer puis reproduire des sons réels, nullement de les fabriquer de toutes pièces.

Pourvu que le vocabulaire utilisé soit relativement limité, on peut malgré tout se servir de ce composant pour construire des phrases à partir d'une série de mots enregistrés une fois pour toutes, dans le cas d'applications du genre horloge parlante ou multimètre vocal.

La carte très économique dont nous allons décrire la réalisation permet d'expérimenter commodément cette technique par simple branchement sur la prise *CENTRONICS* d'un micro-ordinateur, compatible PC ou même familial. Pour en arriver là, il faut évidemment procéder d'abord à l'enregistrement de tous les mots nécessaires, puis faire appel à une logique capable de les appeler dans l'ordre voulu.

Bien des applications se contentent d'un nombre assez limité de mots courts, mais susceptibles d'être assemblés de multiples manières : c'est particulièrement évident dans le cas très riche d'applications de la synthèse vocale de valeurs numériques : heure, résultats de mesures, numéros de téléphone, etc.

Une réflexion relativement simple fait facilement apparaître que 32 ou même 16 mots prononçables chacun en moins d'une seconde suffisent pour ce genre d'exercice.

A raison d'environ 5 kilobits par seconde, ce qui correspond à un son de qualité assez moyenne mais suffisante pour de la parole, une EPROM de type 2764 peut ainsi contenir 16 mots de 750 ms, et une 27128 32 mots. Avec une 27256, on pourrait au choix doubler le nombre de mots ou améliorer la fidélité du son, une 27512 permettant même de faire les deux à la fois.

Le tableau de la figure 3.35 donne un exemple de vocabulaire adapté à cette capacité, et convenant bien à la transmission vocale par radio ou par téléphone de résultats de mesures.

Quelques modifications simples suffiraient pour l'adapter à des applications horaires (horloge parlante).

A chaque mot doit évidemment correspondre une partie bien précise de la mémoire du synthétiseur : un seizième d'une 2764 ou un trente-deuxième d'une 27128. Quatre bits permettent de désigner un mot parmi seize, et cinq bits un mot parmi trente-deux : il s'agira très logiquement des quatre ou cinq lignes d'adresse de poids fort de la mémoire, ce qui laisse

les neuf lignes de poids faible pour permettre à l'UM 5100 de balayer les 512 octets affectés à chaque mot.

0	ZERO	SEIZE
1	UN	VINGT
2	DEUX	TRENTE
3	TROIS	QUARANTE
4	QUATRE	CINQUANTE
5	CINQ	SOIXANTE
6	SIX	CENT
7	SEPT	MILLE
8	HUIT	ET
9	NEUF	VIRGULE
10	DIX	CHANGER
11	ONZE	CALIBRE
12	DOUZE	VOLT
13	TREIZE	MILLI
14	QUATORZE	DEGRÉ
15	QUINZE	MOINS
	2764	2764
	27128	

Fig. 3.35. – Un exemple de "vocabulaire".

Le schéma de la figure 3.36 est basé sur ce principe : l'UM 5100 est monté en configuration lecture, beaucoup plus simple que la configuration enregistrement-lecture utilisée dans nos précédents montages, et communique donc avec la mémoire par les lignes d'adresse A0 à A8.

Les lignes A9 à A12 (2764) ou A13 (27128) rejoignent un circuit d'interface destiné à être raccordé à la prise CENTRONICS d'un micro-ordinateur : ainsi, l'appel de chaque mot se fera par simple impression d'un caractère, la construction des phrases à prononcer pouvant être entièrement gérée par le logiciel.

L'ordinateur fournira donc à la carte un groupe de quatre ou cinq bits désignant le mot à prononcer, validé par une impulsion négative de /STROBE permettant sa mémorisation dans un verrou 74HC373. La carte répondra alors en mettant la ligne BUSY (broche 11) au niveau haut tant que durera l'émission du son, évitant tout risque d'envoi de caractères à un rythme trop rapide.

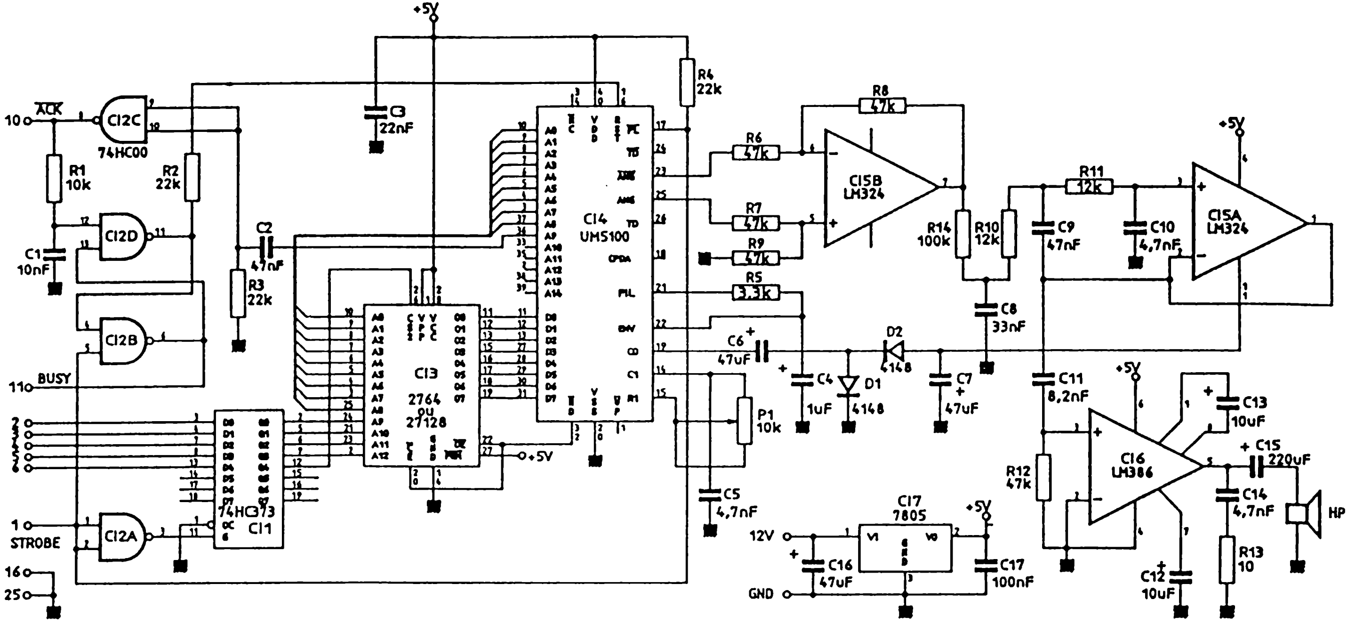


Fig. 3.36. – Schéma du synthétiseur vocal.

Le mot suivant ne pourra donc être appelé que lorsque cette ligne BUSY sera revenue au niveau bas, retour accompagné d'une courte impulsion sur le fil /ACK (broche 10).

La réalisation de la carte ne présente pas de difficulté particulière : il suffit de graver le circuit imprimé simple face de la figure 3.37, et de le câbler en accord avec le plan de la figure 3.38, sans oublier les straps. La prise CENTRONICS, dont le type peut varier d'un ordinateur à l'autre, sera soudée au bout d'un morceau de câble plat à dix conducteurs, de préférence pas trop long.

Un régulateur 7805 étant prévu sur la carte, la tension d'alimentation n'est pas critique : en principe, elle est prévue en 9 ou 12 V.

Cette carte étant uniquement destinée à la reproduction de son, l'enregistrement du vocabulaire nécessaire devra se faire à l'aide de la carte de développement décrite précédemment.

Comme à l'accoutumée, il faudra exécuter la prise de son dans une mémoire ZEROPOWER (RAM à pile lithium) genre MK 48Z08 (SGS-THOMSON), puis transférer son contenu dans une 2764 ou dans la moitié d'une 27128 à l'aide d'un programmateur d'EPROM. Le cas échéant, une seconde prise de son sera effectuée pour remplir la deuxième moitié de la 27128.

Il est commode d'enregistrer en une seule fois la totalité des seize mots qui tiennent ainsi dans 8 K-octets. Afin d'éviter tout décalage, il est souhaitable d'installer un petit métronome sous la forme d'une LED reliée par 1 kΩ à la ligne d'adresse A9 de l'UM 5100 enregistreur (broche 36) : on prononcera le premier mot dès l'appui sur le poussoir d'enregistrement, puis chacun des suivants juste après chaque allumage ou extinction de la LED.

Quelques répétitions doivent suffire pour maîtriser parfaitement cette petite gymnastique...

Le comportement du montage ne pourra être considéré comme significatif que dans la mesure où celui-ci sera relié à un micro-ordinateur. A défaut, on peut relier chacune des entrées de données (2, 3, 4, 5, 6) à la masse ou au +5 V, et mettre brièvement à la masse l'entrée /STROBE (1) pour appeler un mot.

Sous BASIC, un test rapide peut être fait en exécutant quelques instructions

```
LPRINT CHR$(N) ;
```

mais il faut s'attendre à des émissions indésirables de dix et de treize (saut de ligne et retour chariot), selon les machines.

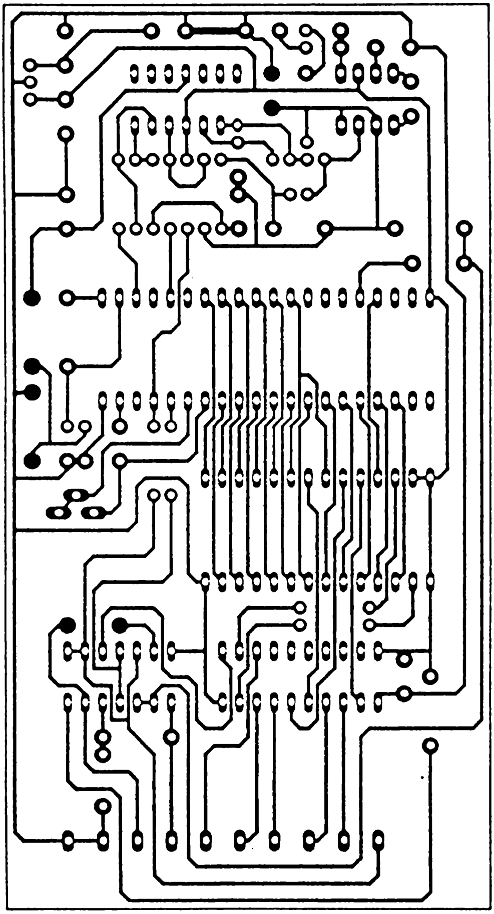


Fig. 3.37. – Circuit imprimé du synthétiseur vocal.

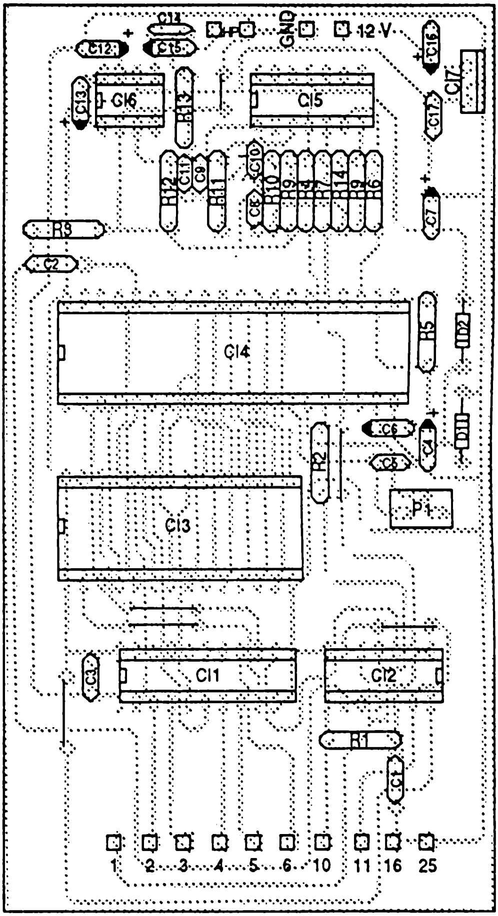


Fig. 3.38. – Plan de câblage du synthétiseur vocal.

Sur compatibles PC, il est possible de dialoguer directement avec le port LPT1 : à l'aide du petit programme de la figure 3.39 : il sert simplement à prononcer les seize premiers mots du vocabulaire de la carte, c'est à dire en principe à compter de 0 à 15.

```
10 FOR F=0 TO 15
20 REM ---- CHIFFRES ----
30 B=INP(889)
40 IF (B AND 128)<>128 THEN 30
50 OUT 888,F
60 OUT 890,1
70 OUT 890,0
80 REM (c)1991 Patrick GUEULLE
90 NEXT F
```

Fig. 3.39. – Logiciel d'essai du synthétiseur vocal.

Par la suite, il conviendra naturellement d'écrire un programme spécifiquement adapté à l'application envisagée : il devra prendre en charge l'assemblage des mots disponibles de toutes les façons nécessaires à l'émission des phrases désirées. Bien entendu, la connexion à un micro-ordinateur n'est qu'un exemple de mise en oeuvre pratique de cette carte, déjà riche d'applications mais pas unique.

Une approche spécialement intéressante pourrait consister à faire appel à un microcontrôleur pour réaliser un système parlant entièrement autonome.

En particulier, d'intéressantes associations pourront être trouvées avec les séquenceurs et contrôleurs programmables décrits au chapitre 2 !

Liste des composants du synthétiseur vocal :

Résistances (5% 1/4 W) :
P1 : potentiomètre ajustable 10 kΩ,
R1 : 10 kΩ,
R2 à R4 : 22 kΩ,
R5 : 3,3 kΩ,
R6 à R9 : 47 kΩ,
R10 et R11 : 12 kΩ,
R12 : 47 kΩ,
R13 : 10 Ω,
R14 (Rx) : 100 kΩ.

Condensateurs :
C1 : 10 nF,
C2 et C9 : 47 nF,
C3 : 22 nF,

C4 : 1 µF 16 V,
C5, C10, C14 : 4,7 nF,
C6, C7, C16 : 47 µF,
C8 : 33 nF,
C11 : 8,2 nF,
C12 et C13 : 10 µF,
C15 : 220 µF 16 V,
C17 : 0,1 µF.

Circuits intégrés :
CI1 : 74HC373,
CI2 : 74HC00,
CI3 : 2764 ou 27128 (voir texte),
CI4 : UM 5100,
CI5 : LM 324,
CI6 : LM 386,
CI7 : 7805 (régulateur positif 5 V).

Autres semiconducteurs :
D1 et D2 : 1N 4148.

④ MONTAGES A CARTES A PUCE

De plus en plus largement utilisées dans toute une gamme d'applications "sensibles" et en particulier comme moyens de paiement ou de contrôle d'accès, les cartes à puce sont souvent présentées comme de véritables forteresses électroniques capables de défendre les secrets qu'elles contiennent envers et contre tout.

C'est à peu près vrai dans le cas des cartes de haut de gamme à microprocesseur incorporé, mais archi-faux pour ce qui est des cartes jetables bon marché que sont notamment les cartes téléphoniques prépayées (télécartes) : nous l'avons amplement démontré dans notre précédent ouvrage *Composants électroniques programmables.*

Le fait que l'on puisse prendre aussi facilement connaissance du contenu d'une carte à puce ne remet cependant nullement en cause sa sécurité : il n'y a finalement rien de secret dans la mémoire d'une télécarte !

Même le fait que l'on puisse tout aussi facilement écrire dans sa mémoire ne remet pas en cause l'inviolabilité de cette véritable monnaie électronique.

Mais après tout, appelons un chat un chat : pour recharger une carte usagée, il faudrait remettre à zéro les bits correspondant aux unités consommées ce qui, dans le cas présent (mémoire EPROM), ne pourrait se faire que par effacement global.

Or, la puce est enrobée dans une résine spéciale, parfaitement opaque aux UV même si elle apparait parfois comme transparente.

Si un pirate astucieux trouvait le moyen d'effacer tout de même la puce, il ne pourrait pas reconstituer la *zone fabricant* dont les 96 bits sont protégés contre l'écriture par la destruction irréversible, en usine, d'un fusible interne : la carte serait définitivement inutilisable !

Cela étant posé, il serait vraiment dommage de ne pas expérimenter avec cette matière première pratiquement inépuisable que sont les télécartes usagées.

Bien sûr, cette technologie est protégée par des brevets en bonne et due forme : pas question donc d'utiliser à des fins commerciales les montages que nous allons décrire, mais rien n'interdit, nous semble-t-il, de les réaliser dans un but expérimental puis de s'en servir à titre privé.

Ajoutons d'ailleurs que chaque télécarte usagée a déjà donné lieu au versement d'une redevance au titre des brevets en question...

UN CONNECTEUR POUR CARTE A PUCE

Pour manipuler commodément avec les cartes à puce, il faut nécessairement disposer d'un connecteur approprié, capable de donner accès aux huit contacts de ce qu'il est convenu d'appeler le *micromodule,* sans abîmer la carte en quoi que ce soit tout en facilitant au maximum son insertion.

Devant l'ampleur du marché des cartes à puce, de multiples fabricants de connecteurs se sont penchés sur la question, et une norme garantissant un certain degré d'interchangeabilité est même en train de se mettre en place.

En principe, ces composants n'étaient guère livrés qu'aux industriels du secteur, et seulement par quantités.

Nous avons cependant pu obtenir qu'un connecteur de bonne qualité soit disponible au détail à un prix abordable chez SELECTRONIC (BP 513, 59022 LILLE CEDEX). Tous les montages qui vont suivre feront bien évidemment appel à cette pièce quasiment incontournable.

Pour faciliter l'incorporation du connecteur dans les réalisations les plus diverses (et en particulier dans les serrures à carte qui exigent le respect de certaines règles de sécurité), nous avons étudié un petit circuit imprimé relais permettant de raccorder à peu près n'importe quoi au connecteur, par l'intermédiaire d'un simple câble plat à dix conducteurs équipé de fiches HE 10.

La figure 4.1 reproduit le tracé de cette carte simple face, dont il faut percer tous les trous à 1 mm à l'exception de ceux de fixation, dont le diamètre sera de 2,5 mm.

Le plan d'implantation de la figure 4.2 montre que deux embases sont prévues pour le câble plat : l'une correspond aux cartes dont les puces sont en position AFNOR comme les télécartes (position dite *excentrée*), tandis que l'autre convient aux puces en position ISO ou *centrée* (cartes de parcmètre PIAF, par exemple). Les connecteurs disposent en effet de deux jeux de balais distincts pour les deux types de cartes.

Ces embases seront tout simplement réalisées à partir de barrettes sécables à double rangée de picots carrés, droits ou mieux coudés. Les picots coudés présentent l'avantage d'interdire l'insertion des fiches HE 10 dans le mauvais sens, grâce à leur bossage détrompeur.

Le câble dont la figure 4.3 donne les détails de réalisation nous servira bientôt à relier ce connecteur de cartes à des montages utilisant des composants tout à fait conventionnels.

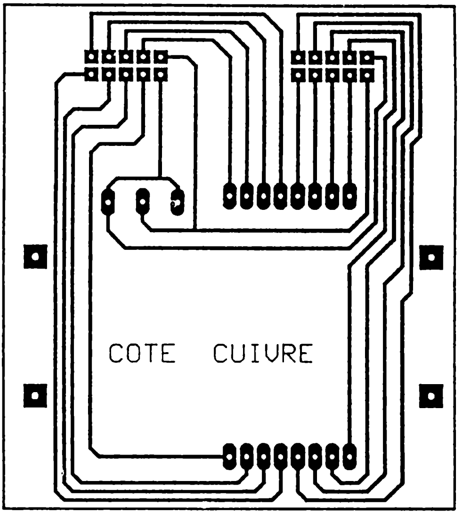


Fig. 4.1. – Circuit imprimé du connecteur de carte.

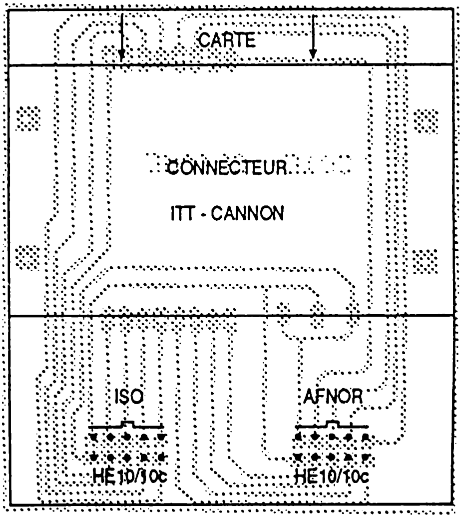


Fig. 4.2. – Plan de câblage du connecteur de carte.

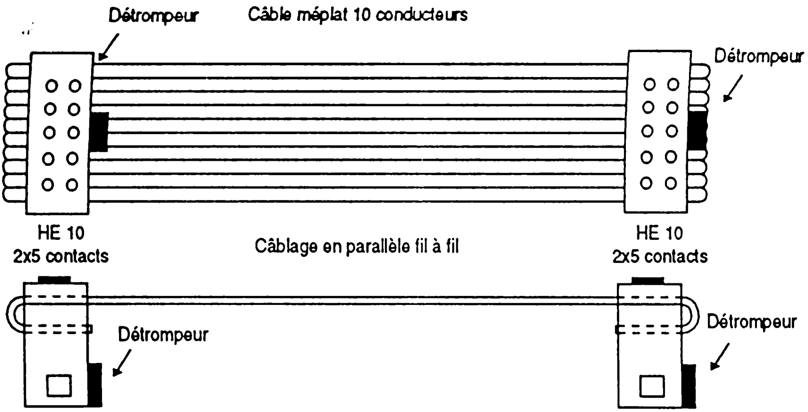


Fig. 4.3. – Réalisation du câble d'interconnexion.

On remarquera que les deux fiches HE 10 doivent être câblées en parallèle fil à fil : on veillera donc à éviter toute inversion au moment du sertissage, qui se fait très facilement au moyen d'un étau à mors lisses ou habillés de petites cornières en aluminium.

Huit des fils du câble correspondent aux contacts de la puce, les deux autres donnant accès au microcontact détectant la présence de la carte dans le connecteur et qui nous servira, par exemple, à mettre sous tension un montage de lecture.

La figure 4.4 donne la correspondance entre la position des contacts sur le micromodule, et les broches des embases HE 10. Les numéros des contacts ne sont pas choisis au hasard, mais sont imposés par la norme ISO relative aux cartes à puce.

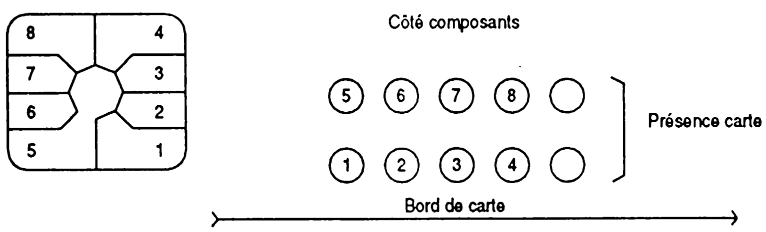


Fig. 4.4. – Correspondance des brochages.

UN MINI-LECTEUR DE TÉLÉCARTES

Outre tout le détail des modalités de lecture et d'écriture dans les télécartes, notre ouvrage *COMPOSANTS ÉLECTRONIQUES PROGRAMMABLES* contient les plans d'un lecteur-programmateur très complet, et bien entendu parfaitement compatible avec les montages qui vont maintenant être décrits.

Cependant, et afin que le présent livre se suffise à lui-même, nous allons d'abord présenter la réalisation d'un lecteur simplifié mais permettant de procéder au *vidage* de n'importe quelle télécarte à l'aide du premier compatible PC venu.

En effet, c'est à partir des données ainsi collectées que nos prochains montages vont être programmés, sans qu'il soit indispensable d'écrire dans les cartes.

Le schéma de la figure 4.5 montre que ce montage est destiné à être branché sur le port d'imprimante parallèle (CENTRONICS) d'un micro-ordinateur compatible PC ou éventuellement d'un autre type.

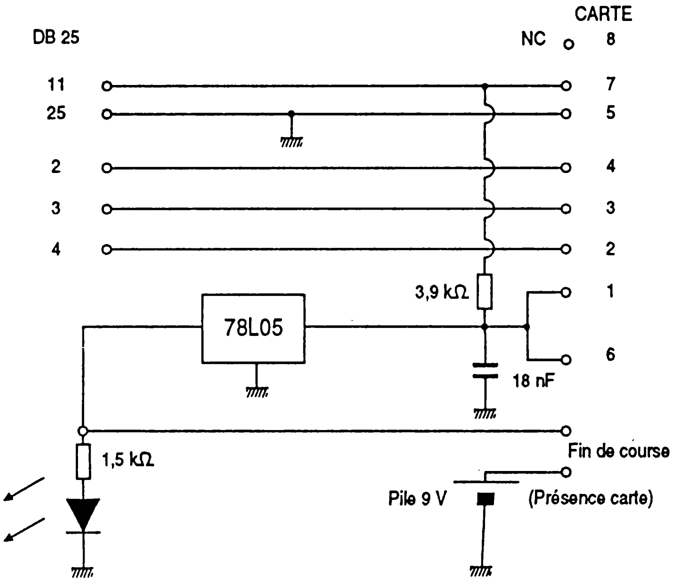


Fig. 4.5. – Schéma du lecteur de cartes.

A part le circuit d'alimentation, produisant du +5 V à partir d'une pile 9 V, le rôle du montage se borne à raccorder cinq broches de la prise DB25 aux balais correspondants du connecteur de carte à puce.

Une diode LED présence carte est prévue pour signaler que la pile est en train de débiter.

Le tableau récapitulatif de la figure 4.6 fournit les correspondances entre brochage de la DB25, désignation des signaux CENTRONICS, numérotation ISO des contacts de la carte à puce, signaux des télécartes, et signaux d'une mémoire EEPROM que nous utiliserons plus loin avec ce même montage (NMC 9306).

DB25	CENTRONICS	ISO	TELECARTE	NMC 9306
11	BUSY	7	S	DO
25	GND	5	GND	GND
2	D0	4	RAZ	CS
3	D1	3	H	SK
4	D2	2	W	Di
V_{CC} (+5 V)		1	V_{CC}	V_{CC}
V_{PP} (+5 V)		6	V_{PP}	N.C.
N.C.		8	FUSIBLE	N.C.

ISO

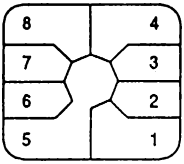


Fig. 4.6. – Correspondance des brochages et des signaux.

Le circuit imprimé de la figure 4.7 a été spécialement dessiné en vue de son incorporation dans un boîtier HEILAND HE222, dont la conception mécanique se prête bien à l'aménagement d'une fente latérale en face de celle du connecteur (il suffit de couper, au disque à tronçonner pour perceuse miniature, juste en face de la rainure de chaque coquille).

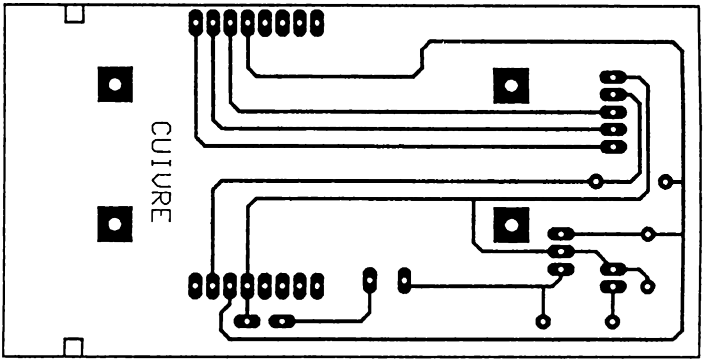


Fig. 4.7. – Circuit imprimé du lecteur de cartes.

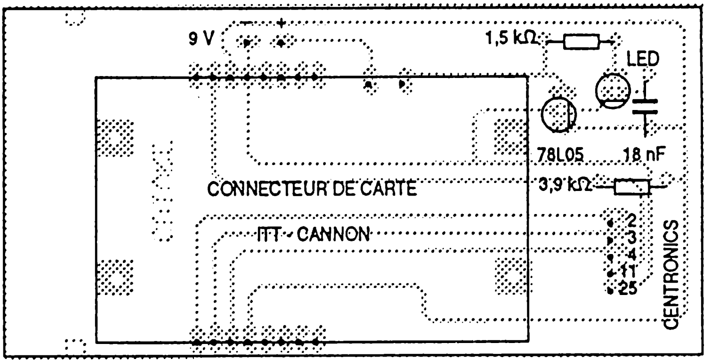
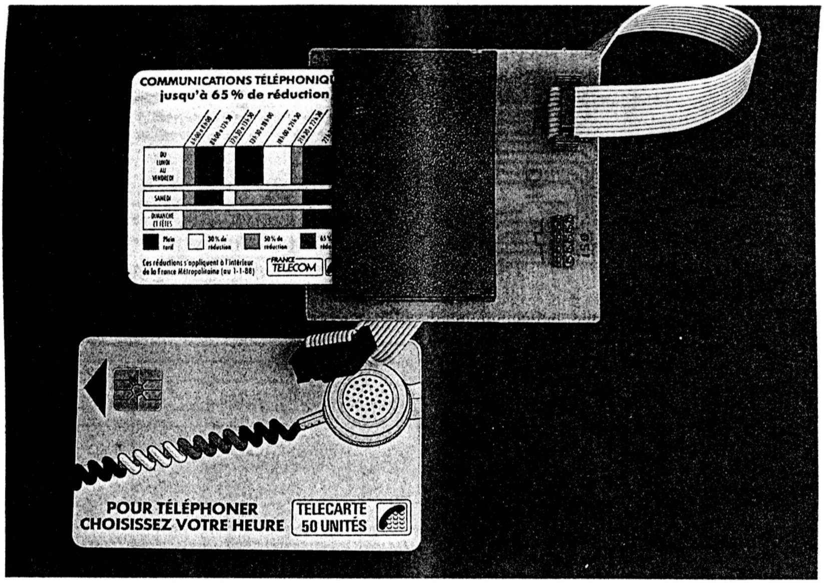


Fig. 4.8. – Plan de câblage du lecteur de cartes.

L'implantation selon la figure 4.8 ne pose pas de problème particulier pourvu que le connecteur de carte soit bien du modèle préconisé (ITT-CANNON).

Une place est réservée dans le boîtier pour la pile 9V, tandis qu'un câble à cinq conducteurs seulement suffit pour raccorder la prise DB25 mâle rejoignant l'ordinateur.



Liste des composants du lecteur de télécartes :

Résistances (5% 1/4 W) :
R1 : 3,9 kΩ,
R2 : 1,5 kΩ.

Condensateur :
C1 : 18 nF

Circuit intégré :
CI1 : 78L05.

Autres semiconducteurs :
D1 : LED rouge.

Divers :
1 connecteur de carte à puce ITT-CANNON.
1 fiche DB 25 mâle,
50 cm de câble à 5 conducteurs,
1 boîtier HEILAND HE222,
1 pile 9 V et son connecteur à clip.

LES LOGICIELS D'EXPLOITATION

Comme d'habitude, nous avons reporté sur le logiciel toutes les contraintes de gestion du système, de façon à aboutir à une partie matérielle aussi simple et universelle que possible.

Les programmes BASIC publiés ici ont été écrits pour des compatibles PC de bas de gamme dont le port d'imprimante est LPT1 :, et dont la fréquence d'horloge ne dépasse pas 8 MHz.

Pour d'autres adresses de port ou d'autres micros (par exemple des AMSTRAD CPC), il faudrait bien entendu modifier ces logiciels en conséquence et donc connaître les adresses d'accès CENTRONICS correspondantes.

On pourra par ailleurs être amené à ralentir artificiellement les programmes en cas d'utilisation sur des machines cadencées à plus de 8 MHz (insertion de quelques boucles FOR-NEXT vides aux endroits stratégiques).

Le court programme TELECART.BAS de la figure 4.9 permet de visualiser les 256 bits que renferme une télécarte, selon la présentation de la figure 4.10 : huit lignes de huit groupes de quatre bits.

```
10 REM ---- TELECART ----
20 E=&H379:S=&H378
30 CLS
40 OUT S,0
50 PRINT"Insérer carte, puis presser ENTER"
60 INPUT Z$
70 BEEP:FOR T=0 TO 400:NEXT T
80 OUT S,250:OUT S,248
90 FOR F=1 TO 8
100 FOR G=1 TO 8
110 FOR H=1 TO 4
120 OUT S,249
130 D=INP(E):D= (D AND 128)
140 IF D= 128 THEN PRINT"0";
150 IF D<>128 THEN PRINT"1";
160 OUT S,251
170 NEXT H
180 PRINT" ";:NEXT G
190 PRINT:NEXT F
200 PRINT:GOTO 40
210 REM (c)1992 Patrick GUEULLE
```

Fig. 4.9. – Le logiciel d'examen de télécartes.

```
1 0 1 1   1 1 1 1   0 0 0 0   0 0 1 1   1 0 0 0   0 1 1 0   0 1 1 0   1 0 0 1
1 0 1 1   0 0 1 1   0 0 1 1   0 1 1 0   0 1 1 0   0 1 0 0   0 1 0 1   1 1 0 1
1 1 0 0   1 0 1 1   0 0 0 0   1 1 0 1   0 0 0 1   0 0 0 0   0 0 0 0   0 1 1 0
1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1   0 0 0 0
0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0   1 1 1 1   1 1 1 1
```

Fig. 4.10. – Exemple de contenu d'une télécarte épulsée.

Les trois premières lignes correspondent aux 96 bits de la *zone fabricant*, accessible uniquement en lecture et probablement unique pour chaque télécarte en circulation. Les cinq dernières représentent la zone des unités, dans laquelle il reste possible de transformer n'importe quel **0** en **1**, mais de façon irréversible : de quoi programmer toutes sortes de codes personnels pour renforcer encore la sécurité des applications que nous allons développer.

Le programme CRECARTE-BAS de la figure 4-11 sert pour sa part à lire le crédit de n'importe quelle télécarte. L'expérience montre en effet que beaucoup de cartes de récupération ne sont pas vides, et il serait vraiment dommage de ne pas en profiter !

Il affichera sans équivoque :

TELECARTE de 50 (ou 120, voire 40) unités
Consommation : tant d' UTC

et selon le cas :

IL RESTE tant d' UTC

ou, hélas :

CREDIT EPUISE

On pourra également lire :

TELECARTE NON UTILISABLE

ou bien :

Carte anormale !

Il faut entendre par *télécarte non utilisable* une carte à puce dans laquelle le logiciel ne trouve pas le code spécifique de l'application télécarte : il peut s'agir d'une carte de cinéma, de stationnement, bancaire ou PASTEL, mais aussi d'une carte sans puce ou d'une télécarte insérée à l'envers ou défectueuse (puce grillée ou mauvais contact). En fait, ce message correspond assez exactement au *carte non utilisable dans cet appareil* des cabines publiques.

```
10 REM ----- CRECARTE.BAS -----
20 E=&H379:S=&H378
30 OUT S,0:CLS
40 PRINT"Insérer carte, puis presser ENTER"
50 INPUT Z$:CLS
60 OUT S,250:OUT S,248
70 FOR F=1 TO 8:GOSUB 350:NEXT F
80 A$=""
90 FOR F=9 TO 16
100 GOSUB 350:A$=A$+D$:NEXT F
110 IF A$<>"00000011" THEN PRINT"TELECARTE NON UTILISABLE":END
120 FOR F=17 TO 91:GOSUB 350:NEXT F
130 A$=""
140 FOR F=92 TO 96
150 GOSUB 350:A$=A$+D$:NEXT F
160 C$="de type inconnu"
170 IF A$="10011" THEN C$="de 120 unités":C=120
180 IF A$="00110" THEN C$="de 50 unités":C=50
190 IF A$="00101" THEN C$="de 40 unités":C=40
200 PRINT"TELECARTE ";C$;"  ";
210 N=0
220 FOR F=97 TO 106:GOSUB 350:NEXT F
230 FOR F=107 TO 248
240 GOSUB 350
250 IF D$="1" THEN N=N+1
260 NEXT F
270 PRINT"Consommation: ";N;" UTC"
280 A$=""
290 FOR F=249 TO 256
300 GOSUB 350:A$=A$+D$:NEXT F
310 IF A$="11111111" THEN PRINT"CREDIT EPUISE"
320 IF A$<>"11111111" THEN PRINT"IL RESTE ";C-N;" UTC":BEEP
330 IF N>C THEN PRINT"Carte anormale!":BEEP
340 PRINT:PRINT:GOTO 40
350 OUT S,249
360 D=INP(E):D= (D AND 128)
370 IF D=128 THEN D$="0"
380 IF D<>128 THEN D$="1"
390 OUT S,251
400 RETURN
410 REM (c)1991 Patrick GUEULLE
```

Fig. 4.11. – Le logiciel de lecture du crédit des télécartes.

En principe, une télécarte réellement défectueuse est remboursable auprès de n'importe quelle agence commerciale de France Télécom (c'est le cas d'une proportion non négligeable des cartes abandonnées dans les cabines : vous avez donc toutes vos chances !).

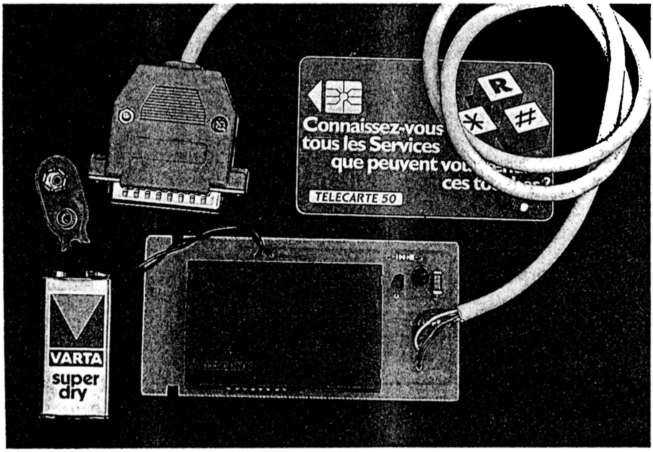
Carte anormale signale par contre une télécarte en état de marche, mais dont les unités ont été consommées de manière fantaisiste. Dans une telle situation, il est à conseiller de procéder à un examen plus fin du contenu de la carte avec TELECART-BAS, car de tels spécimens sont assez rares et pourraient bien acquérir une certaine valeur de collection (au même titre que les timbres-poste mal imprimés) lorsque notre appareil sera suffisamment répandu chez les *télécartistes*.

Un bip sonore est d'ailleurs émis lorsque le résultat de la lecture est digne d'intérêt : carte contenant encore des unités, non utilisable, ou anormale.

Mais c'est le logiciel LECARTE-BAS de la figure 4.12 qui va vraiment ouvrir la porte à toutes sortes d'applications pratiques. Comme TELECART-BAS, il affiche le contenu de la carte, mais il construit surtout un fichier disque qui, muni de l'extension .CAR, nous servira bientôt à programmer les lecteurs chargés de reconnaître uniquement certaines cartes.

```
10 REM ----- LECARTE.BAS -----
20 CLS:PRINT"Nom à donner au fichier .CAR ?"
30 INPUT N$:IF N$="" THEN N$="CARTE"
40 N$=N$+".CAR"
50 OPEN N$ FOR OUTPUT AS#1
60 E=&H379:S=&H378
70 OUT S,0
80 PRINT"Insérer carte, puis presser ENTER"
90 INPUT Z$:CLS
100 OUT S,250:OUT S,248
110 FOR F=1 TO 8
120 FOR G=1 TO 8
130 FOR H=1 TO 4
140 OUT S,249
150 D=INP(E):D= (D AND 128)
160 IF D=128 THEN PRINT#1,"0 ";:PRINT"0";
170 IF D<>128 THEN PRINT#1,"1 ";:PRINT"1";
180 OUT S,251
190 NEXT H
200 PRINT#1," ";:PRINT" ";:NEXT G
210 PRINT#1,:PRINT:NEXT F
220 PRINT:PRINT:PRINT"Le fichier ";N$;" est disponible"
230 PRINT:PRINT:END
240 REM (c)1991 Patrick GUEULLE
```

Fig. 4.12. – Le logiciel de lecture de télécartes.



UNE SERRURE ÉLECTRONIQUE A TÉLÉCARTE

Deux approches différentes sont envisageables pour utiliser les télécartes de récupération comme clefs dans une serrure électronique.

La plus simple consiste à contrôler le contenu d'une carte non modifiée : lire ainsi la totalité de la zone fabricant permettrait selon toute probabilité de construire une serrure ne répondant qu'à une seule et unique carte, pas nécessairement usagée d'ailleurs.

En ne lui faisant lire qu'une partie de cette zone de 96 bits, on pourrait par contre faire en sorte que la serrure reconnaisse plus d'une carte : certaines similitudes partielles se retrouvent en effet d'une carte à l'autre.

En lisant les bits 89 à 96, on pourrait par exemple n'accepter que les cartes de 40 ou de 120 unités, nettement moins répandues que celles de 50.

En s'intéressant par contre aux huit premiers bits, on pourrait définir au moins six *familles* différentes de cartes.

Une autre approche consiste à personnaliser la zone des bits inutilisés : une carte de 50 unités épuisée peut par exemple facilement héberger dix ou onze groupes de huit bits, dont chacun peut prendre 256 valeurs différentes.

On pourrait donc imaginer de construire dix serrures réagissant chacune à l'un de ces groupes de huit bits : il suffirait alors d'inscrire le bon code au bon endroit dans chaque carte devant pouvoir ouvrir la serrure en question.

Mieux, chaque carte pourrait être programmée pour pouvoir ouvrir une seule, plusieurs, ou toutes les serrures, et ce pas nécessairement en une seule fois : une carte déjà émise pourrait fort bien être mise à jour de façon à pouvoir ouvrir une ou plusieurs serrures de plus, de même qu'il serait tout aussi facile de lui retirer l'autorisation d'ouvrir telle ou telle serrure : il suffirait pour cela de surcharger avec des **1** tout ou partie des **0** du code correspondant !

On le voit, les perspectives d'applications de cette technique sont immenses...

Le lecteur pour PC que nous venons de décrire pourrait bien évidemment être piloté par un logiciel qui le ferait fonctionner en *serrure informatique.*

On imagine cependant mal d'immobiliser un micro-ordinateur simplement pour gérer l'ouverture d'une porte (à moins qu'il ne faille tenir à jour une liste des allées et venues basée sur l'identité des cartes...).

Mais nous avons décrit au chapitre 2 un petit automate à usage général qui se prête justement très bien à être programmé pour cette application !

La figure 4.13 montre comment une carte à puce peut facilement être interfacée avec cet automate : il suffit en fait de reconstituer le complément logique du signal de sortie de la carte pour que le programme puisse tester aussi bien les niveaux hauts que les niveaux bas pendant qu'il va faire défiler les bits à lire, grâce à des signaux judicieusement dirigés sur les entrées de commande de la carte.

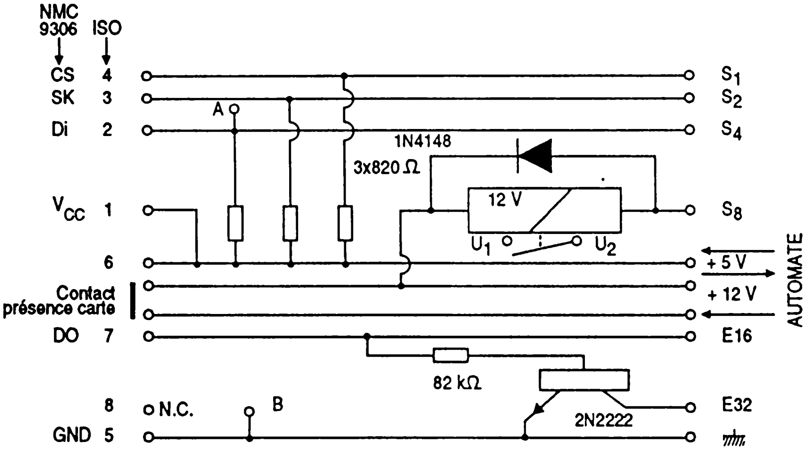


Fig. 4.13. – Schéma de l'interface pour l'automate.

La serrure complète se compose donc de trois modules :

- l'automate programmable (figures 2.6 et 2.7)
- le connecteur de carte (figures 4.1 et 4.2)
- le circuit d'interface (figures 4.14 et 4.15)

Le connecteur de carte sera placé là où les contraintes d'utilisation l'exigent (souvent en extérieur, dans un boîtier étanche), l'automate programmable non loin de là où la serrure doit faire son effet, et le circuit d'interface à proximité de l'automate : surtout pas avec le connecteur de carte, exposé à toutes les malveillances !

On raccordera le connecteur au circuit d'interface par un câble plat conforme au modèle de la figure 4.3, tandis que l'automate pourra être relié à l'interface à l'aide de simples fils soudés.

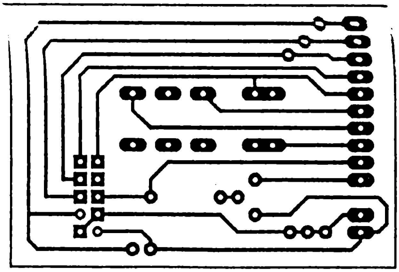


Fig. 4.14. – Circuit imprimé de l'interface.

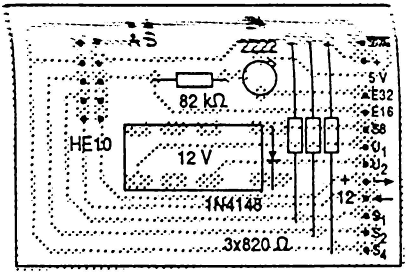
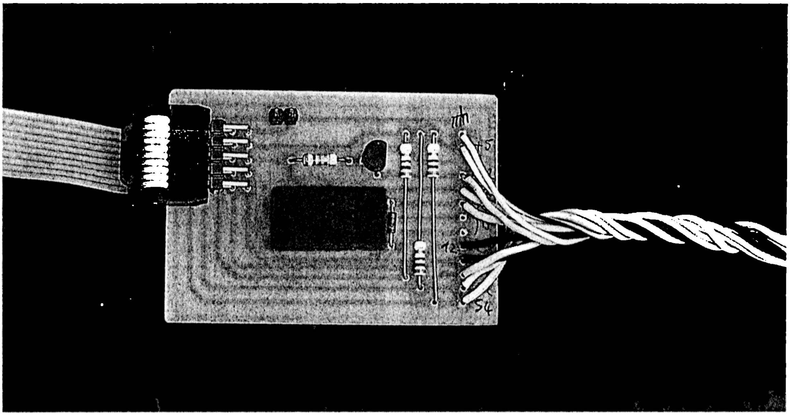


Fig. 4.15. – Plan de câblage de l'interface.



L'automate se trouve alimenté à travers le contact détectant la présence de la carte, et peut donc si nécessaire fonctionner sur pile ou batterie. Il démarre l'exécution de son programme dès l'introduction de la carte.

Les 100 premières adresses de l'EPROM pourront contenir l'octet 00000111 (7 en décimal) : cette courte attente permet d'éliminer les rebondissements des contacts de la carte.

Suivent un octet 00000101 (5 en décimal) et trois octets 00000111, le tout servant à remettre à zéro le compteur d'adresses de la carte.

Pour chaque bit devant être sauté sans lecture, on doit ensuite programmer les deux octets 00000110 (6 en décimal) et 00000100 (4 en décimal).

Pour chaque bit qu'il faut vérifier comme étant à 0 (E1 à 0), il faut par contre programmer 00010110 (22 en décimal) puis 00000100.

Pour chaque bit à vérifier comme étant à 1 (E2 à 0), il faut cette fois programmer 00100110 (38 en décimal) et 00000100.

Enfin, le programme se termine sur un octet 10001111 (139 en décimal).

Ce dernier octet ne sera atteint que si le contenu de la carte est bien conforme à ce que l'on attend : un niveau bas est alors dirigé sur la sortie S8 pour faire coller le relais, tandis que l'exécution s'arrête là car jamais l'entrée E128 ne pourra passer à 0 puisqu'elle est en l'air.

De même, au premier bit non conforme à ce qui est attendu, l'exécution du programme s'arrête et l'octet commandant le collage du relais ne sera jamais atteint.

Bien entendu, tout s'arrête aussi dès que l'on retire la carte de son connecteur, coupant ainsi l'alimentation générale.

Il n'en faut pas davantage pour commander une gâche électrique dans des conditions de sécurité de fort bon niveau, mais à bien peu de frais.

Reste maintenant à programmer l'EPROM avec les octets permettant de reconnaître spécifiquement telle ou telle carte.

En fait, compte tenu de la capacité généreuse de la 2764 utilisée, on peut loger deux programmes distincts dans la même mémoire, et choisir l'un ou l'autre par simple commutation d'un cavalier (H-L) : cela peut être utile, par exemple, si on désire autoriser certaines cartes le jour et pas la nuit, ou en semaine mais pas le week-end.

A défaut de micro-ordinateur, il serait éventuellement possible de programmer une EPROM à la main à partir des indications qui viennent d'être fournies : il faudrait alors examiner manuellement le contenu de la carte à lire.

Mais voyons plutôt comment opérer avec l'aide de l'outil informatique !

Le logiciel SERRCART.BAS, listé à la figure 4.16, opère à partir d'un fichier .CAR reproduisant le contenu d'une carte préalablement lue avec LECARTE.BAS.

Il suffit de lui préciser le groupe de bits à tester pour qu'il construise automatiquement un fichier de même nom mais muni de l'extension .ROM, contenant les octets à programmer dans l'EPROM de l'automate.

Ces octets sont codés en décimal, mais le programme ROMTOBIN.BAS de la figure 3.7 est là pour permettre d'obtenir le fichier binaire dont ont besoin certains programmateurs d'EPROM.

Evidemment, on peut programmer l'automate pour qu'il lise n'importe quel groupe de bits de la carte, de 2 à 256, selon ce que l'on souhaite obtenir en pratique : à chaque utilisateur de décider !

```
10 REM ----- SERRCART.BAS -----
20 CLS:PRINT"Nom du fichier .CAR à utiliser ?"
30 INPUT N$:C$=N$+".CAR":R$=N$+".ROM"
40 CLS:PRINT"La carte contient 256 bits: 1 à 256"
50 OPEN C$ FOR INPUT AS #1
60 PRINT:PRINT"----------------------------------------"
70 PRINT:PRINT"         Zone  fabricant: 1 à 96"
80 OPEN R$ FOR OUTPUT AS #2
90 PRINT:PRINT"         Zone des unités: 97 à 256"
100 PRINT:PRINT"----------------------------------------"
110 PRINT:PRINT"Premier bit à tester ?"
120 INPUT D
130 IF D<1 OR D>255 THEN CLS:GOTO 110
140 PRINT:PRINT"Dernier bit à tester ?"
150 INPUT F
160 IF F<=D OR F>256 THEN CLS:GOTO 110
170 CLS:PRINT"----- Création fichier ";R$;" en cours -----"
180 FOR G=1 TO 100
190 PRINT#2,7;
200 NEXT G
210 PRINT#2,5;7;7;7;
220 FOR G=1 TO D-1
230 INPUT#1,B
240 PRINT#2,6;4;
250 NEXT G
260 FOR G=D TO F
270 INPUT#1,B
280 IF B=0 THEN PRINT#2,22;4;
290 IF B=1 THEN PRINT#2,38;4;
300 NEXT G
310 PRINT#2,139;
320 CLOSE
330 CLS:PRINT"----- LE FICHIER ";R$;" EST DISPONIBLE -----
340 END
350 REM (c)1992 Patrick GUEULLE
```

Fig. 4.16. – Le logiciel de programmation de l'automate.

Liste des composants de la serrure à télécarte :

Résistances (5% 1/4 W) :
R1 à R3 : 820 Ω,
R4 : 82 kΩ.

Transistors :
T1 : 2N 2222 ou BC 107.

Autres semiconducteurs :
D1 : 1N 4148.

Divers :
1 relais 12 V 2RT DIL,
1 barrette sécable 2 x 5 picots coudés,
1 module connecteur de carte à puce,
1 module automate programmable,
1 alimentation 12 à 15 V continu.

UN ADAPTATEUR POUR CLEF A PUCE

Spécialisée dans l'électronique horlogère et la micromécanique, la Société SEFEA d'Annemasse possède les deux compétences nécessaires au développement d'applications originales autour des circuits intégrés présentés en micromodules pour cartes à puce.

Sa clef à mémoire n'est rien d'autre qu'une robuste clef en plastique (ABS), sur laquelle est rapporté un micromodule de carte à puce. Parmi les circuits intégrés disponibles sous cette forme, le plus courant est le TS 1200 de SGS-THOMSON, rencontré justement dans les télécartes (EPROM série de 256 bits).

Le connecteur correspondant est un solide guide en aluminium anodisé ou en plastique, usiné avec précision et muni d'une *bascule porte-contacts*. Ce n'est que tout à fait en fin de course de la clef que les balais *atterrissent* sur le micromodule, puis que le contact de détection de clef se ferme.

L'ensemble est particulièrement résistant et fiable, souvent plus qu'un système à carte électriquement équivalent.

Par rapport à l'utilisation de télécartes de récupération, l'usage de ces clefs présente la particularité que le fusible de protection en écriture des 96 premiers bits (la *zone fabricant*) est intact lors de la livraison.

Il en résulte des possibilités de personnalisation plus larges, au prix toutefois d'un risque de duplication des clefs.

Electriquement parlant, la clef à TS 1200 est strictement équivalente à une télécarte, à ceci près que le fusible est intact et que la mémoire est pratiquement vide.

Rien ne s'oppose donc à ce que l'on utilise les mêmes circuits de programmation et de lecture, moyennant bien sûr une adaptation de connectique.

Notons cependant que le connecteur de clef est muni de dix picots dont le brochage est compatible avec notre câble de la figure 4.3 : la serrure que nous venons de décrire pourra donc être équipée indifféremment d'un connecteur de carte ou de clef !

En ce qui concerne notre mini-lecteur ou le lecteur-enregistreur décrit dans *COMPOSANTS ÉLECTRONIQUES PROGRAMMABLES*, il est commode d'éviter toute modification en passant par l'intermédiaire d'une *fausse carte* réalisée en circuit imprimé de 8/10 de millimètre.

Le stratifié époxy présensibilisé nécessaire est un produit standard de CIF (Circuit Imprimé Français), la dimension de 100 × 160 mm en simple

face (Ref- AAB16) permettant largement de tirer deux cartes conformes au tracé de la figure 4.17.

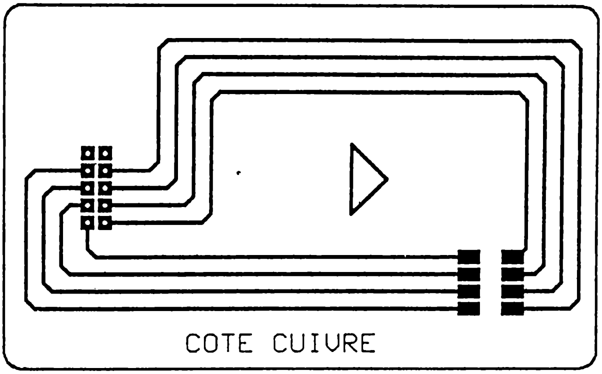


Fig. 4.17. – Le tracé de la "fausse carte" en circuit imprimé.

Bien entendu, le côté cuivre de ce circuit imprimé correspond au côté *contacts* des cartes à puce, et le sens d'introduction dans le lecteur devra être respecté en conséquence.

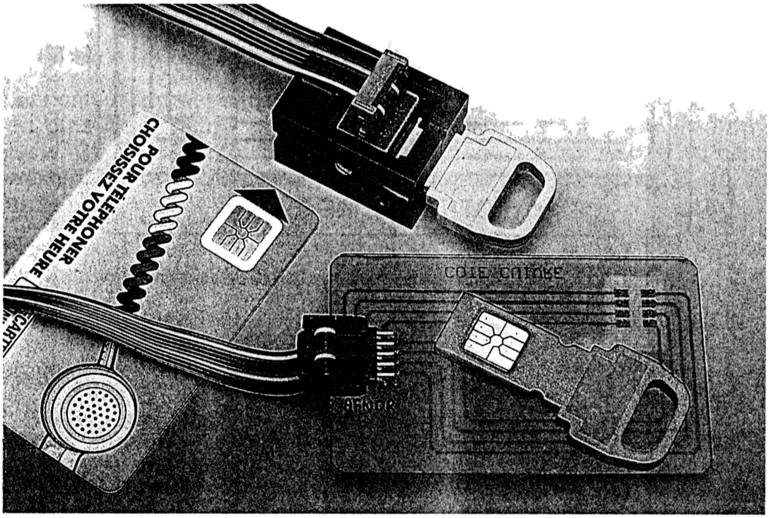
Il reste donc à assurer la liaison fil à fil entre les contacts du connecteur de clef et ceux de la fausse carte, toujours à l'aide du câble universel de la figure 4.3.

La fiche HE10 sera branchée sur le connecteur SEFEA de telle façon que le câble parte à l'opposé du trou d'introduction de la clef (autrement dit vers l'arrière).

Grâce à cette adaptation, l'appareil doit fonctionner avec une clef à TS 1200 (référencée MK 10 chez SEFEA) exactement comme il le ferait avec une télécarte, à ceci près que la lecture d'une clef neuve se traduit par un seul bit à 1 parmi 255 bits à 0 : cela suffit pour empêcher toute imitation frauduleuse des cartes téléphoniques !

Au niveau de la serrure elle-même, rappelons qu'il est souhaitable que seul le connecteur soit exposé à e possibles malveillances : son câble, éventuellement prolongé, devra donc rejoindre un endroit sûr dans lequel seront installés l'adaptateur et l'automate.

L'orifice du connecteur de clef étant nettement plus vaste que celui d'un connecteur de carte, on pourra avantageusement l'équiper, en extérieur, d'un volet de protection rappelé par un ressort ou par son propre poids. Cela évitera la pénétration d'eau en cas d'intempéries, et limitera les introductions de corps étrangers auxquelles sont plus ou moins exposées toutes les entrées de serrure.



UNE SERRURE ÉLECTRONIQUE À EEPROM

Par rapport aux mémoires EPROM que contiennent les télécartes, les EEPROM présentent l'avantage de pouvoir être reprogrammées, en tout ou partie, un très grand nombre de fois.

Cela peut être utile lorsque l'on doit modifier fréquemment les prérogatives des cartes ou clefs employées dans des applications de contrôle d'accès.

En revanche, la duplication et la falsification deviennent plus faciles : s'il est quasiment impossible d'imiter la *zone fabricant* d'une télécarte, par contre rien n'est plus simple que de recopier une EEPROM *non sécurisée* dans une autre !

Nous allons maintenant décrire une variante à EEPROM de notre serrure à télécarte : elle sera bien sûr plus souple d'emploi, mais son degré de sécurité sera un peu moindre : à chacun de choisir en fonction de son application !

UNE CARTE À PUCE À EEPROM

Bien que le principe même de la carte à puce consiste à loger des circuits intégrés dans ses 0,76 mm d'épaisseur, il n'est pas ridicule de chercher à mettre en carte des composants classiques présentés en boîtiers DIL ou CMS.

En effet, la plupart des lecteurs ou connecteurs de carte n'*avalent* guère plus de la moitié de celle-ci, ce qui laisse une importante surface susceptible de tolérer une certaine surépaisseur.

Ce principe, largement employé pour la confection de cartes de test ou de prototypes de circuits destinés à un futur encartage, permet d'utiliser toutes sortes de circuits intégrés plus courants que les micromodules employés par les fabricants de cartes.

A vrai dire, une carte à puce n'est rien d'autre qu'une plaquette isolante sur laquelle sont rapportés des contacts plats reliés à un circuit électronique : cela s'apparente finalement d'assez près à un circuit imprimé...

De là à imaginer de réaliser des cartes à puce à partir de plaquettes d'époxy cuivré de même épaisseur, il n'y a qu'un pas que nous avons d'ailleurs déjà franchi en fabriquant notre *fausse carte* adaptatrice.

De son côté, l'EEPROM NMC 9306 est un composant facilement disponible et peu coûteux, doté de fonctionnalités qui le destinent à toutes sortes d'applications très comparables à celles des cartes à puce : mémorisation de données, contrôle d'accès, protection de logiciels, etc.

La figure 4.18 reproduit son schéma synoptique.

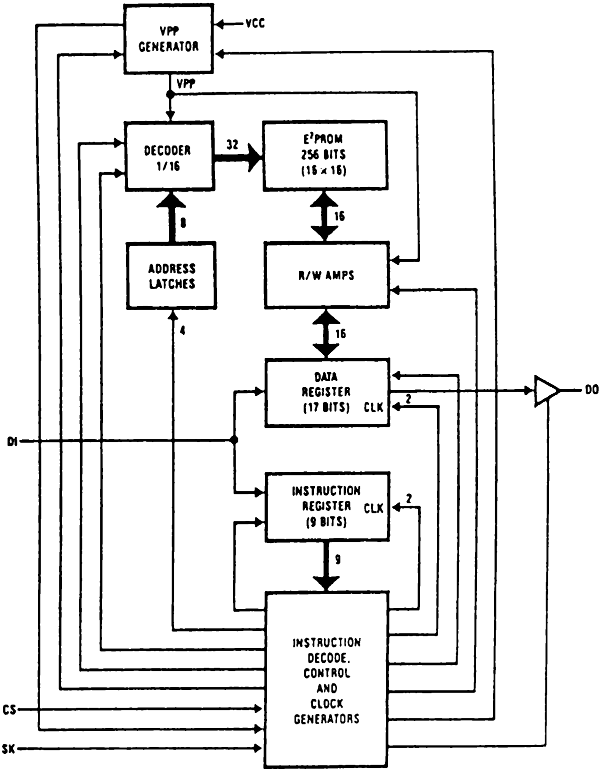


Fig. 4.18. - Organisation interne de la NMC 9306.

C'est une mémoire de 256 bits, comme les télécartes, mais à cette différence près que son accès est aléatoire et non plus séquentiel : elle est organisée en seize registres de seize bits chacun, qui peuvent être écrits, lus, ou effacés individuellement.

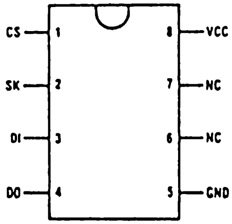
S'agissant d'une EEPROM, cette mémoire est en effet réinscriptible au moins dix mille fois par registre, tandis que la rétention des données est de l'ordre d'une bonne dizaine d'années sans alimentation.

Comme pour les cartes à puce, son contenu est accessible par l'entremise d'un bus série, en l'occurence un bus MICROWIRE (marque déposée de National Semiconductor).

Six connexions suffisent donc pour alimenter le circuit et pour établir le dialogue, comme en témoigne le brochage reproduit à la figure 4.19 :

CS	Chip Select (sélection du boîtier)
SK	Horloge de transmission série
DI	Data Input (entrée des données)
DO	Data Output (sortie des données)
Vcc	Alimentation (tension unique +5 V)
GND	Masse

Dual-In-Line Package



Top View

Pin Names

CS	Chip Select
SK	Serial Data Clock
DI	Serial Data Input
DO	Serial Data Output
V_{CC}	Power Supply
GND	Ground

Fig. 4.19. – Brochage de l'EEPROM NMC 9306.

Deux broches sont donc inutilisées sur le boîtier DIP à 8 pattes dans lequel est présentée la NMC9306N de NS, tandis que les broches DI et DO peuvent éventuellement être réunies lorsqu'une seule entrée-sortie de données est nécessaire.

Contrairement au circuit TS1200 équipant par exemple les télécartes, les données ne sont pas les seules informations circulant sur les lignes d'entrée-sortie : les nécessités de l'accès aléatoire aux seize registres imposent l'échange d'instructions et d'adresses.

Le tableau de la figure 4.20 récapitule le jeu d'instructions disponible, dont le format est le suivant :

- un bit de départ (startbit) à 1
- un code opératoire (opcode) à 4 bits
- une adresse de registre sur 4 bits
- s'il y a lieu, seize bits de données

Instruction	SB	Op Code	Address	Data	Comments
READ	1	10xx	A3A2A1A0		Read register A3A2A1A0
WRITE	1	01xx	A3A2A1A0	D15–D0	Write register A3A2A1A0
ERASE	1	11xx	A3A2A1A0		Erase register A3A2A1A0
EWEN	1	0011	xxxx		Erase/write enable
EWDS	1	0000	xxxx		Erase/write disable
ERAL	1	0010	xxxx		Erase all registers
WRAL	1	0001	xxxx	D15–D0	Write all registers

NMC9306E has 7 instructions as shown. Note that MSB of any given instruction is a "1" and is viewed as a start bit in the interface sequence. The next 8 bits carry the op code and the 4-bit address for 1 of 16, 16-bit registers.
X is a don't care state.

Fig. 4.20. – Le jeu d'instructions de la NMC 9306.

La figure 4.21 détaille les chronogrammes applicables à chacune de ces instructions, qu'il importe de respecter avec précision : faute de ramener la ligne CS à zéro entre deux opérations de lecture, par exemple, la seconde donnerait pour résultat l'adresse du registre et non les données qu'il contient !

Notons qu'il est possible d'autoriser (par l'instruction EWEN) ou d'interdire (par EWDS) l'écriture, et évidemment aussi l'effacement, de la totalité des registres.

La mémoire NMC9306 pourrait fort bien être montée sur des micromodules pour cartes à puce, en appliquant la technique *chip on board.*

Pour notre part, nous allons utiliser la version courante en boîtier DIP et la monter sur une *fausse carte* réalisée en époxy 8/10.

Reste à définir une correspondance entre le brochage de la mémoire et celui de la carte.

Indépendamment de toute normalisation, il faut évidemment que cette carte puisse être programmée et lue avec le matériel que nous avons développé pour réutiliser les télécartes usagées.

Le tracé de la figure 4.22 satisfait à cet impératif, et respecte la position de puce définie par la norme AFNOR (contacts excentrés). Cependant, la zone correspondant aux contacts ISO (centrés) a été laissée vide de cuivre afin d'éviter tout risque de court-circuit en cas d'insertion de la carte dans un connecteur multinormes.

L'implantation de la figure 4.23 parle d'elle-même : le boîtier DIP sera inséré de façon conventionnelle, côté non cuivré, et soudé côté cuivre, l'ergot de positionnement étant dirigé vers le centre de la carte.

Il n'est normalement pas nécessaire de prévoir un support, à moins que l'on n'envisage d'utiliser cette fausse carte comme simple adaptateur pour programmer des NMC9306 qui seront ensuite transportées sur un autre circuit imprimé.

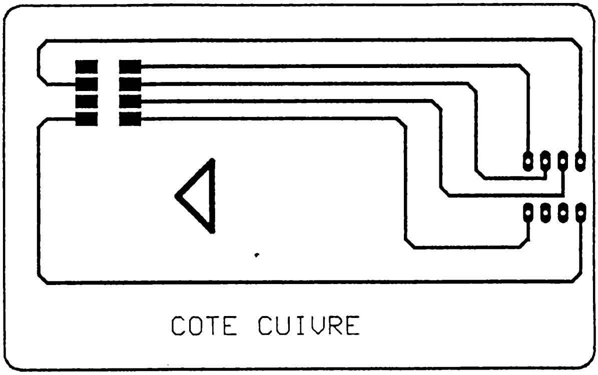


Fig. 4.22. – Circuit imprimé de la carte à puce à EEPROM.

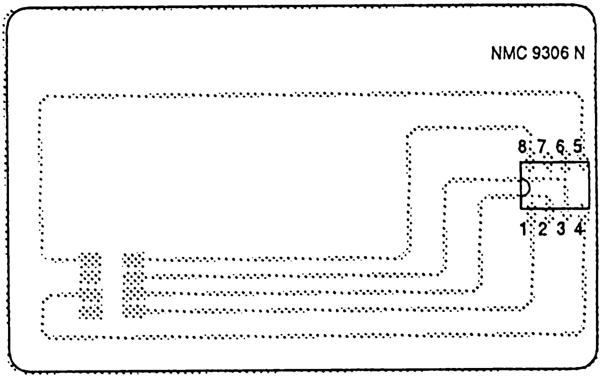


Fig. 4.23. – Plan de câblage de la carte à puce à EEPROM.

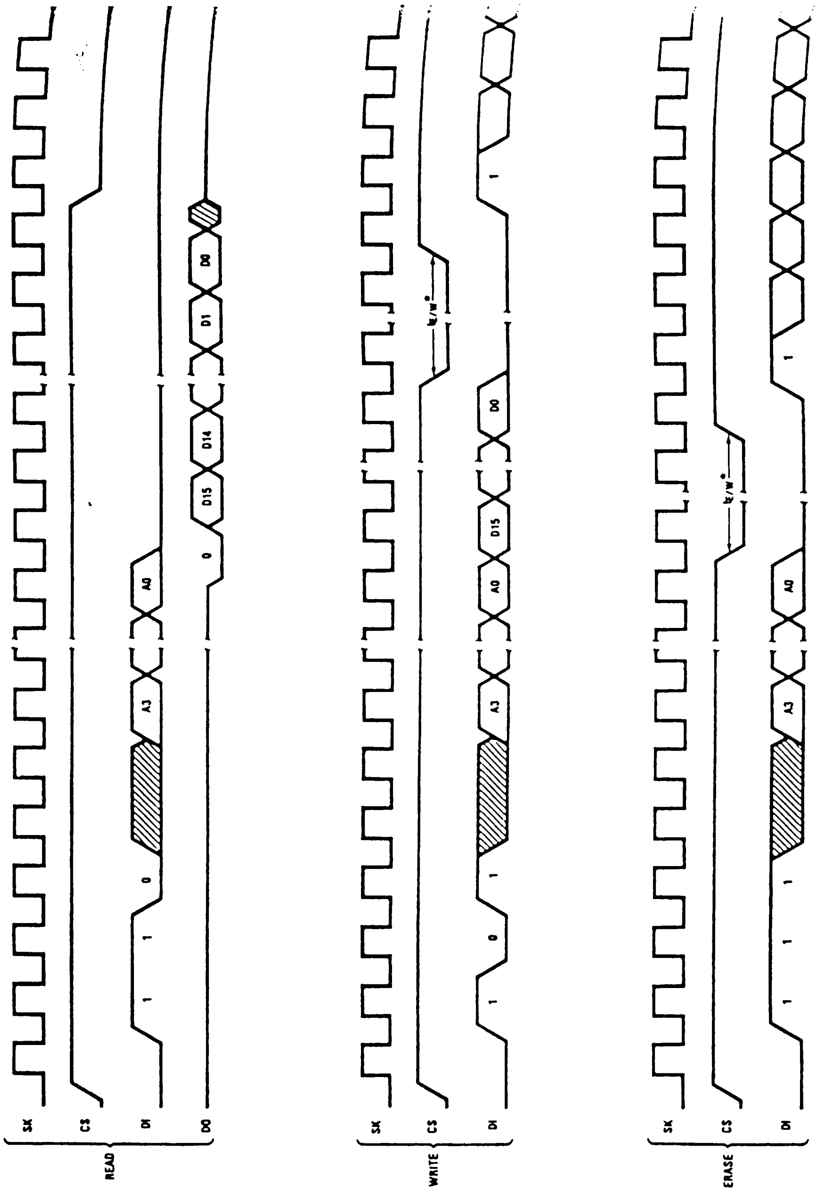
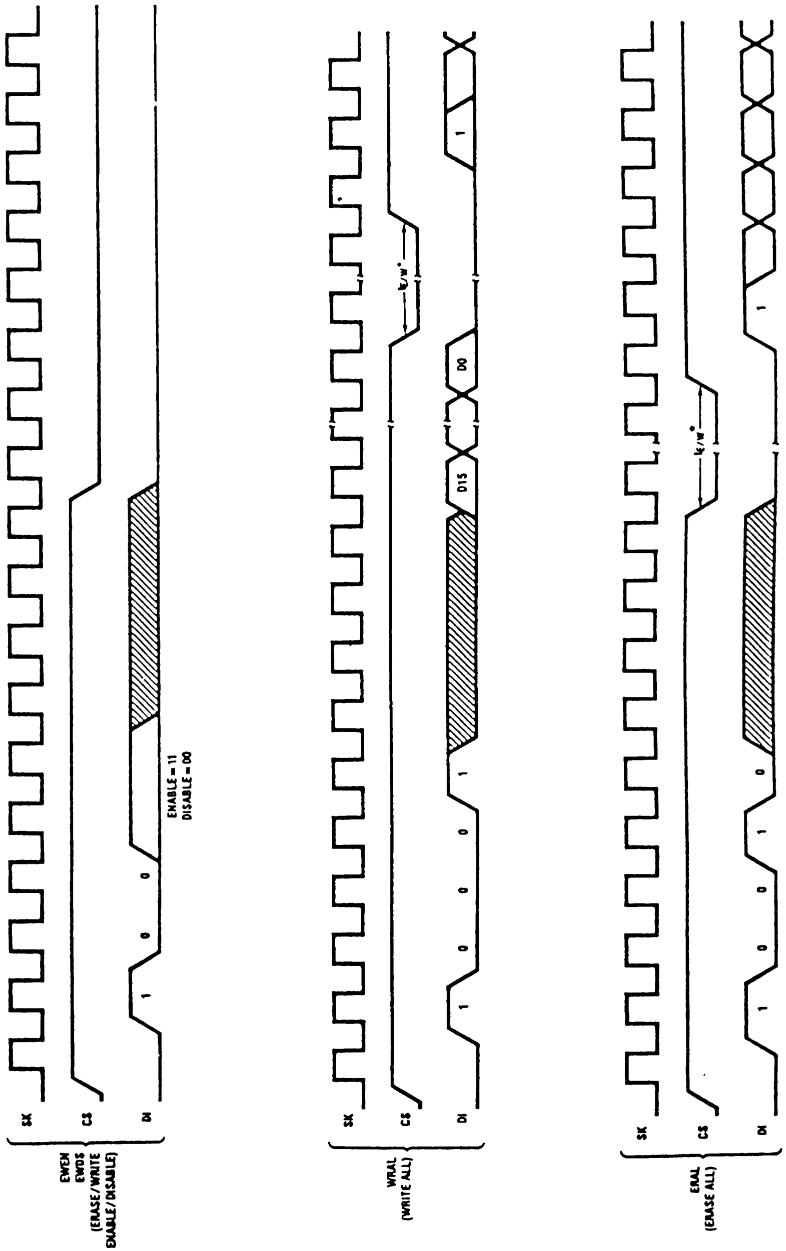


Fig. 4.21. – Les chronogrammes de la NMC 9306.


SK
CS
DI
EWEN
EWDS
(ERASE/WRITE
ENABLE/DISABLE)
ENABLE = 11
DISABLE = 00
WRAL
(WRITE ALL)
D15
D0
ERAL
(ERASE ALL)

UNE CLEF À EEPROM

Une autre solution, particulièrement élégante, consiste à fabriquer une clef à partir du connecteur *Memory Key* spécialement développé à cet effet par AMPHENOL. La figure 4.24 décrit la conception mécanique de cette pièce, à laquelle correspond un réceptacle approprié.

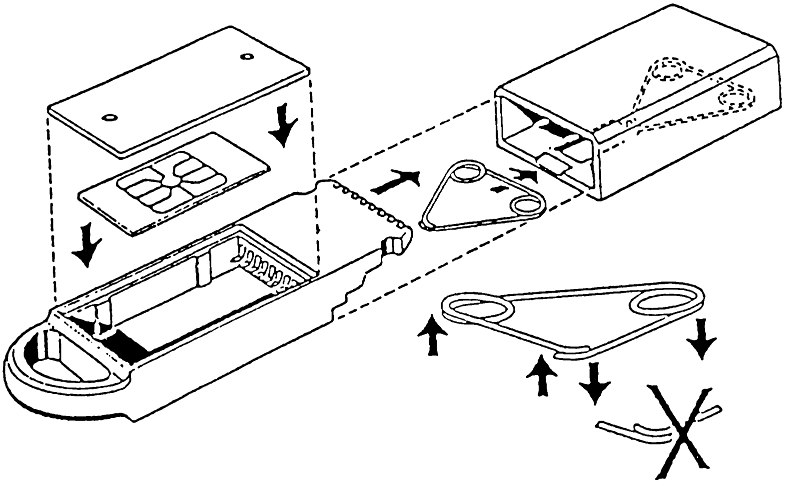


Fig. 4.24. – La "Memory Key" d'AMPHENOL.

Une place suffisante est disponible dans le corps de la clef pour loger soit un micromodule découpé dans une télécarte (grâce au petit circuit imprimé de la figure 4.25), soit une NMC 9306 en boîtier DIP, que l'on soudera alors côté cuivre du petit circuit de la figure 4.26 (après avoir replié les broches de la mémoire).

Dans les deux cas, il faudra employer du stratifié de 8/10, qui se glisse tout juste sous les huit plots à souder disponibles à l'intérieur de la clef et qu'il faudra souder très soigneusement.

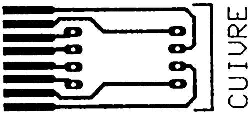


Fig. 4.25. – Circuit imprimé pour puce de télécarte.

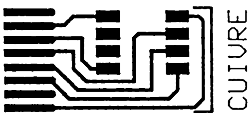


Fig. 4.26. – Circuit imprimé pour boîtier DIP.

Le réceptacle, pour sa part, sera monté sur un circuit imprimé gravé conformément au tracé de la figure 4.27 : équipé de deux rangées de cinq picots coudés, il est entièrement compatible avec tout notre système basé sur le câble de la figure 4.3.

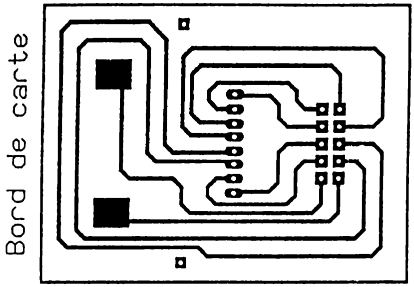


Fig. 4.27. – Circuit imprimé pour le réceptacle.

Pour les applications nécessitant un contact détectant la présence de la clef, une place est prévue pour souder une petite ampoule ILS côté cuivre. Il suffit alors de placer un petit aimant à l'intérieur de la clef pour parachever la compatibilité avec nos précédents montages, et en particulier avec l'interface de la figure 4.15.

LES LOGICIELS D'EXPLOITATION

Comme d'habitude, nous avons reporté sur le logiciel toutes les contraintes de gestion du système, de façon à aboutir à une partie matérielle aussi simple et universelle que possible.

Les programmes BASIC publiés ici ont été écrits pour des compatibles PC de bas de gamme dont le port d'imprimante est LPT1:, et dont la fréquence d'horloge ne dépasse pas 8 MHz.

Pour d'autres adresses de port ou d'autres micros (par exemple des AMSTRAD CPC), il faudrait bien entendu modifier ces logiciels en conséquence et donc connaître les adresses d'accès CENTRONICS correspondantes.

On pourra par ailleurs être amené à ralentir artificiellement les programmes en présence de PC plus rapides, afin de ne pas transgresser les chronogrammes imposés.

Le logiciel de base (READ.BAS listé à la figure 4.28) se charge de lire la totalité du contenu d'une EEPROM, soit 256 bits.

Le résultat de cette opération est visualisé à l'écran sous la forme de seize groupes de seize bits, et simultanément enregistré sur disque sous la forme d'un fichier baptisé EEPROM.CAR (que l'on pourra renommer par la suite).

Le format de ce fichier (bits 1 ou 0 en *mode texte*) est compatible avec celui utilisé par nos précédents logiciels dédiés aux télécartes. Seule la répartition des 256 bits est différente : en groupes de quatre pour les télécartes, et en blocs de seize pour les EEPROM, afin de respecter leur organisation interne.

```
10 REM ----- READ -----
20 S=888:E=889
30 OUT S,0:CLS
40 PRINT"Insérer carte, puis presser ENTER"
50 INPUT Z$:CLS
60 OPEN "eeprom.car" FOR OUTPUT AS #1
70 DATA 0,1,1,0,0,0
80 FOR L=0 TO 7
90 FOR C=0 TO 1
100 R=(2*L)+C
110 B8=0:B4=0:B2=0:B1=0
120 IF R>7 THEN B8=1:R=R-8
130 IF R>3 THEN B4=1:R=R-4
140 IF R>1 THEN B2=1:R=R-2
150 B1=R
160 RESTORE
170 FOR F=1 TO 6
180 READ B
190 GOSUB 340
200 NEXT F
210 B=B8:GOSUB 340
220 B=B4:GOSUB 340
230 B=B2:GOSUB 340
240 B=B1:GOSUB 340
250 FOR F=1 TO 16
260 B=0:GOSUB 340
270 IF D=128 THEN PRINT"0";:PRINT#1,"0 ";
280 IF D<>128 THEN PRINT"1";:PRINT#1,"1 ";
290 NEXT F
300 OUT S,0:OUT S,1
310 PRINT#1," ";:PRINT" ";:NEXT C
320 PRINT#1,:PRINT:NEXT L
330 END
340 B=4*B
350 OUT S,B+1
360 OUT S,B+3
370 OUT S,B+1
380 D=INP(E):D=D AND 128
390 RETURN
400 REM (c)1992 Patrick GUEULLE
```

Fig. 4.28. – Le logiciel de lecture.

Il en résulte que l'on pourra parfaitement recopier une télécarte dans une EEPROM et vice versa, bien que les deux types de cartes ne soient nullement interchangeables au niveau de leurs applications respectives : pas question bien sûr de téléphoner avec une EEPROM !

Le programme ERASE.BAS (listé à la figure 4.29) sert à effacer la totalité d'une EEPROM, opération qui était bien évidemment impossible avec les EPROM OTP que sont les télécartes.

```
10 REM ----- ERASE -----
20 S=888:E=889
30 RESTORE:CLS
40 OUT S,0
50 PRINT"Insérer la carte à effacer, et presser ENTER"
60 INPUT Z$:OUT S,1
70 CLS:PRINT"----- EFFACEMENT EN COURS -----"
80 DATA 0,1,0,0,1,1,1,1,1,1
90 DATA 0,1,0,0,1,0,1,1,1,1
100 N=10:GOSUB 160
110 GOSUB 260
120 N=10:GOSUB 160
130 GOSUB 260
140 CLS:PRINT"EFFACEMENT TERMINE":BEEP
150 END
160 FOR F=1 TO N
170 READ B:GOSUB 210
180 NEXT F
190 OUT S,0
200 RETURN
210 B=4*B
220 OUT S,B+1
230 OUT S,B+3
240 OUT S,B+1
250 RETURN
260 OUT S,0
270 FOR T=0 TO 20:NEXT T
280 RETURN
290 REM (c)1992 Patrick GUEULLE
```

Fig. 4.29. – Le logiciel d'effacement.

Un tel effacement général remet à 1 tous les bits de la mémoire.

La manœuvre consiste à exécuter successivement une instruction EWEN et une instruction ERAL (voir figure 4.20). Il en résulte que ce programme peut effacer une EEPROM même si on l'a protégée par une instruction EWDS.

Si on souhaite que le programme ERASE respecte les mémoires protégées par EWDS, il faut neutraliser son instruction EWEN : pratiquement, cela revient à supprimer les lignes 80, 100, et 110.

Le programme WRITE.BAS (figure 4.30) accomplit l'opération symétrique de READ, c'est-à-dire le transfert dans la mémoire du contenu d'un fichier .CAR.

```
10 REM ----- WRITE -----
20 S=888:E=889:CLS
30 PRINT"NOM DU FICHIER .CAR A TRANSFERER ?"
40 INPUT N$:N$=N$+".CAR"
50 OPEN N$ FOR INPUT AS #1
60 CLS:OUT S,0
70 PRINT"Insérer la carte, puis presser ENTER"
80 INPUT Z$:CLS
90 PRINT"----- PROGRAMMATION EN COURS -----"
100 OUT S,1
110 DATA 0,1,0,0,1,1,0,0,0,0
120 FOR F=1 TO 10
130 READ B:GOSUB 430
140 NEXT F
150 GOSUB 480
160 OUT S,1
170 DATA 0,1,0,1,1,1
180 FOR Q=0 TO 15
190 R=Q
200 RESTORE 170
210 B8=0:B4=0:B2=0:B1=0
220 IF R>7 THEN B8=1:R=R-8
230 IF R>3 THEN B4=1:R=R-4
240 IF R>1 THEN B2=1:R=R-2
250 B1=R
260 FOR F=1 TO 6
270 READ B:GOSUB 430
280 NEXT F
290 B=B8:GOSUB 430
300 B=B4:GOSUB 430
310 B=B2:GOSUB 430
320 B=B1:GOSUB 430
330 FOR F=1 TO 16
340 INPUT#1,B:GOSUB 430
350 NEXT F
360 GOSUB 480
370 OUT S,1
380 NEXT Q
400 CLS:PRINT"PROGRAMMATION TERMINEE"
410 PRINT"Retirer la carte":BEEP
420 END
430 B=4*B
440 OUT S,B+1
450 OUT S,B+3
460 OUT S,B+1
470 RETURN
480 OUT S,0
490 FOR T=0 TO 300:NEXT T
500 RETURN
510 REM (c)1992 Patrick GUEULLE
```

Fig. 4.30. – Le logiciel d'écriture.

N'oublions pas, d'ailleurs, que le format retenu pour ces fichiers .CAR permet de les *travailler* à volonté à l'aide d'un simple éditeur de texte...

En général, on exécutera ERASE avant WRITE, mais on peut éventuellement surcharger l'ancien contenu de la mémoire : on se souviendra alors que l'on ne peut, à ce stade, que transformer des **1** en **0** et non l'inverse : c'est exactement le contraire de ce que l'on pouvait faire avec les télécartes.

Là encore, une instruction EWEN est exécutée avant toute chose, ce qui permet d'écrire même dans une EEPROM protégée par EWDS.

Si l'on souhaite respecter la protection, il faut supprimer les lignes 110 à 160.

Le programme PROTECT.BAS, justement, sert à protéger une EEPROM par exécution d'une instruction EWDS : pas spécialement pour empêcher l'exécution de WRITE ou ERASE, mais plutôt après programmation et avant mise en service de la mémoire dans un autre montage (clef électronique, dongle, mémoire de configuration, etc.)

Il est listé à la figure 4.31.

```
10 REM ----- PROTECT -----
20 S=888:E=889
30 RESTORE:CLS
40 OUT S,0
50 PRINT"Insérer la carte à protéger, et presser ENTER"
60 INPUT Z$:OUT S,1
70 DATA 0,1,0,0,0,0,0,0,0,0
80 N=10:GOSUB 120
90 GOSUB 220
100 CLS:PRINT"CARTE PROTEGEE":BEEP
110 END
120 FOR F=1 TO N
130 READ B:GOSUB 170
140 NEXT F
150 OUT S,0
160 RETURN
170 B=4*B
180 OUT S,B+1
190 OUT S,B+3
200 OUT S,B+1
210 RETURN
220 OUT S,0
230 FOR T=0 TO 300:NEXT T
240 RETURN
250 REM (c)1992 Patrick GUEULLE
```

Fig. 4.31. – Le logiciel de protection d'écriture.

Le programme CHECK.BAS (figure 4.32) permet de comparer bit à bit le contenu d'une EEPROM avec celui d'un fichier .CAR, par exemple celui à partir duquel on vient de la programmer, ou bien celui à partir duquel on l'avait programmée jadis.

Les bits conformes sont représentés sur l'écran par des tirets, et les bits erronés par des astérisques. Par ailleurs, un compte-rendu en clair s'imprime au terme du contrôle.

```
10 REM ----- CHECK -----
20 S=888:E=889:K=0
30 OUT S,0:CLS
40 PRINT"Nom du fichier .CAR de référence ?"
50 INPUT N$:N$=N$+".CAR"
60 OPEN N$ FOR INPUT AS #1
70 CLS:PRINT"Insérer carte, puis presser ENTER"
80 INPUT Z$:CLS
90 DATA 0,1,1,0,0,0
100 FOR L=0 TO 7
110 FOR C=0 TO 1
120 R=(2*L)+C
130 B8=0:B4=0:B2=0:B1=0
140 IF R>7 THEN B8=1:R=R-8
150 IF R>3 THEN B4=1:R=R-4
160 IF R>1 THEN B2=1:R=R-2
170 B1=R
180 RESTORE
190 FOR F=1 TO 6
200 READ B
210 GOSUB 420
220 NEXT F
230 B=B8:GOSUB 420
240 B=B4:GOSUB 420
250 B=B2:GOSUB 420
260 B=B1:GOSUB 420
270 FOR F=1 TO 16
280 B=0:GOSUB 420
290 INPUT#1,X
300 IF D=128 AND X=0 THEN PRINT"-";
310 IF D=128 AND X=1 THEN PRINT"*";:K=1
320 IF D<>128 AND X=1 THEN PRINT"-";
330 IF D<>128 AND X=0 THEN PRINT"*";:K=1
340 NEXT F
350 OUT S,0:OUT S,1
360 PRINT" ";:NEXT C
370 PRINT:NEXT L
380 PRINT:PRINT:PRINT
390 IF K=0 THEN PRINT"CONFORME"
400 IF K=1 THEN PRINT"NON conforme":BEEP
410 END
420 B=4*B
430 OUT S,B+1
440 OUT S,B+3
450 OUT S,B+1
460 D=INP(E):D=D AND 128
470 RETURN
480 REM (c)1992 Patrick GUEULLE
```

Fig. 4.32. – Le logiciel de vérification.

Le programme BLANK.BAS, enfin, réalise le test de virginité d'une mémoire neuve ou que l'on vient d'effacer : comme précédemment, les bits bien effacés (à 1) sont représentés par des tirets et les bits non effacés (à 0) par des astérisques.

Un compte-rendu très clair est également affiché en fin de test.

Ce programme est listé à la figure 4.33.

```
10 REM ----- BLANK -----
20 S=888:E=889:A=0
30 OUT S,0:CLS
40 PRINT"Insérer carte, puis presser ENTER"
50 INPUT Z$:CLS
60 DATA 0,1,1,0,0,0
70 FOR L=0 TO 7
80 FOR C=0 TO 1
90 R=(2*L)+C
100 B8=0:B4=0:B2=0:B1=0
110 IF R>7 THEN B8=1:R=R-8
120 IF R>3 THEN B4=1:R=R-4
130 IF R>1 THEN B2=1:R=R-2
140 B1=R
150 RESTORE
160 FOR F=1 TO 6
170 READ B
180 GOSUB 360
190 NEXT F
200 B=B8:GOSUB 360
210 B=B4:GOSUB 360
220 B=B2:GOSUB 360
230 B=B1:GOSUB 360
240 FOR F=1 TO 16
250 B=0:GOSUB 360
260 IF D=128 THEN PRINT"*";:A=1
270 IF D<>128 THEN PRINT"-";
280 NEXT F
290 OUT S,0:OUT S,1
300 PRINT" ";:NEXT C
310 PRINT:NEXT L
320 PRINT:PRINT:PRINT
330 IF A=1 THEN PRINT "NON ";:BEEP
340 PRINT"VIERGE"
350 END
360 B=4*B
370 OUT S,B+1
380 OUT S,B+3
390 OUT S,B+1
400 D=INP(E):D=D AND 128
410 RETURN
420 REM (c)1992 Patrick GUEULLE
```

Fig. 4.33. – Le logiciel de test de virginité.

Il va maintenant s'agir d'exploiter les cartes et clefs à EEPROM ainsi programmées sur notre serrure, initialement conçue pour fonctionner avec des télécartes.

Côté matériel, la compatibilité est d'ores et déjà acquise : la carte à puce à EEPROM peut être directement insérée dans le connecteur de carte, tandis que le circuit imprimé du réceptacle pour clef AMPHENOL est directement interchangeable avec le connecteur de clef SEFEA.

Mais l'EPROM de l'automate doit évidemment être programmée d'une façon complètement différente puisque les protocoles de communication à mettre en oeuvre sont radicalement dissemblables.

```
10 REM ----- SERE2PR.BAS -----
20 CLS:PRINT"Nom du fichier .CAR à utiliser ?"
30 INPUT N$:C$=N$+".CAR":R$=N$+".ROM"
40 OPEN C$ FOR INPUT AS #1
50 CLS:PRINT"NUMERO DU REGISTRE A TESTER ? (0 à 15)"
60 INPUT R:IF R>15 THEN 50
70 B8=0:B4=0:B2=0:B1=0:X=16*R
80 IF R>7 THEN B8=1:R=R-8
90 IF R>3 THEN B4=1:R=R-4
100 IF R>1 THEN B2=1:R=R-2
110 B1=R
120 CLS:PRINT"----- Création fichier ";R$;" en cours -----"
130 OPEN R$ FOR OUTPUT AS #2
140 FOR F=1 TO 128
150 PRINT#2,7;
160 NEXT F
170 GOSUB 380:GOSUB 380:GOSUB 380
180 GOSUB 380:GOSUB 390:GOSUB 390
190 GOSUB 380:GOSUB 380:GOSUB 380
200 B=B8:GOSUB 400
210 B=B4:GOSUB 400
220 B=B2:GOSUB 400
230 B=B1:GOSUB 400
240 IF X=0 THEN 280
250 FOR F=1 TO X
260 INPUT#1,B
270 NEXT F
280 FOR F=1 TO 16
290 INPUT#1,B
300 IF B=0 THEN PRINT#2,6;4;22;
310 IF B=1 THEN PRINT#2,6;4;38;
320 NEXT F
330 PRINT#2,255;
340 CLOSE
350 CLS:PRINT"LE FICHIER ";R$;" EST DISPONIBLE"
360 PRINT"pour programmer la 2764":BEEP
370 END
380 PRINT#2,6;4;6;:RETURN
390 PRINT#2,2;0;2;:RETURN
400 IF B=0 THEN GOSUB 380
410 IF B=1 THEN GOSUB 390
420 RETURN
430 REM (c)1992 Patrick GUEULLE
```

Fig. 4.34. – Le logiciel de programmation de l'automate.

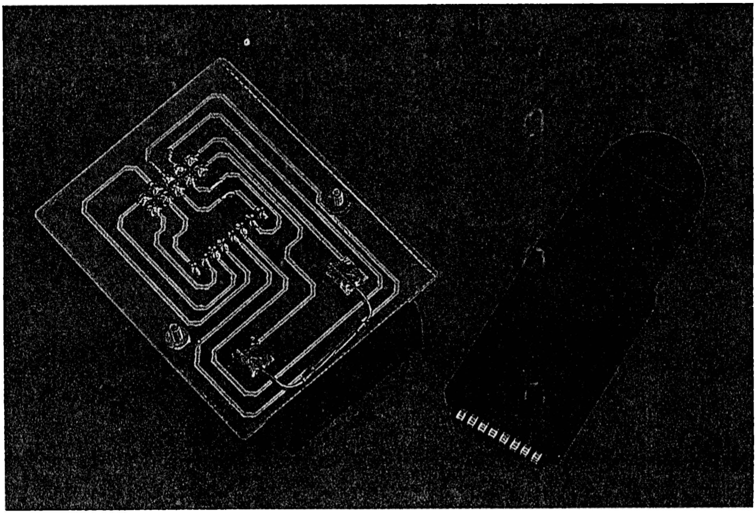
Le logiciel SERE2PR.BAS de la figure 4.34 est prévu, tout comme SERR-CART.BAS, pour construire le fichier .ROM nécessaire à la programmation de la 2764, cela à partir du fichier .CAR reproduisant le contenu de l'EEPROM qui devra servir de clef.

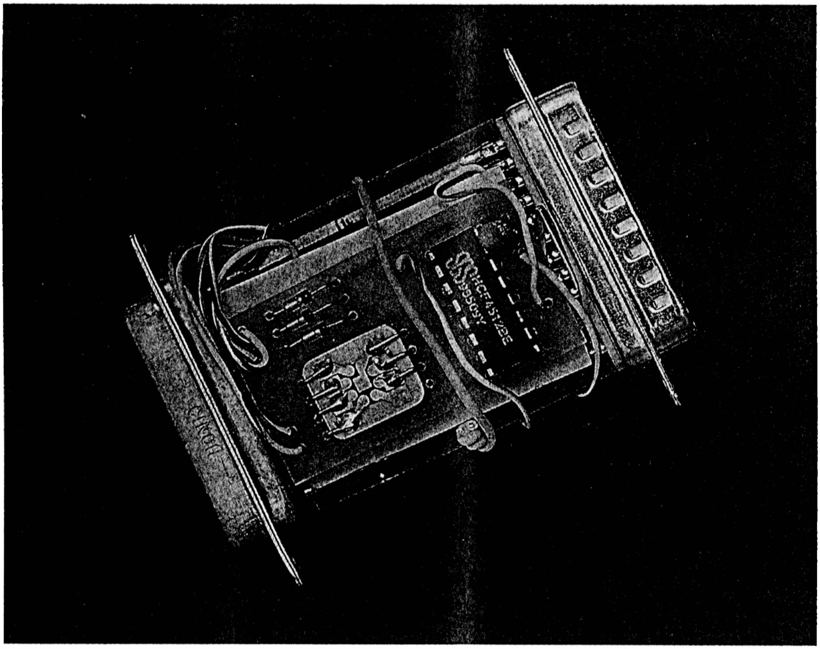
Rappelons que ce fichier .ROM se compose de valeurs décimales, mais qu'il peut être converti en fichier binaire à l'aide de notre utilitaire ROM-TOBIN.BAS listé à la figure 3.7.

La serrure fonctionnera par lecture d'un des seize registres de la mémoire, c'est-à-dire de seize bits : c'est bien suffisant en matière de nombre de combinaisons possibles, tout en permettant de gérer très facilement jusqu'à seize serrures avec un même jeu de cartes ou de clefs.

A tout moment, on pourra donner ou retirer à n'importe quelle carte ou clef l'autorisation de commander n'importe laquelle des seize serrures, en modifiant le contenu du registre correspondant.

Et bien entendu, l'opération est entièrement réversible, ce qui ne serait pas le cas avec une télécarte à EPROM dont on aurait surchargé quelques bits.





UN "*DONGLE*" A PUCE DE TÉLÉCARTE

Parmi les divers moyens utilisés pour protéger les logiciels contre les copies illicites, les *dongles* arrivent en bonne position.

Le secret est généralement bien gardé quant au contenu de ces petites boîtes noires que l'on intercale le plus souvent entre l'ordinateur et le câble d'imprimante.

Le schéma que nous vous proposons ici est particulièrement original : en effet, sa pièce maîtresse n'est autre qu'une puce récupérée sur une télécarte épuisée !

Un avantage certain de ce choix est la gratuité totale de ce composant, mais ce n'est pas le seul : nous allons voir qu'il offre aussi une sécurité de fort bon niveau, bien que le schéma du *dongle* ne soit désormais plus du tout secret...

Dans l'immense majorité des cas, un *dongle* est un petit boîtier réunissant deux embases DB25 (une mâle et une femelle), et prévu pour venir s'intercaler entre le PC et le câble d'une imprimante parallèle. Certains modèles peuvent également être installés sur un port série.

La plupart du temps, une résine d'enrobage est coulée dans le boîtier afin d'interdire l'examen de son schéma interne et donc son éventuelle reproduction.

Bien évidemment, la présence du dongle ne doit modifier en rien le fonctionnement de l'imprimante, et encore moins celui du PC et des autres logiciels dont il est équipé.

Dans la pratique, beaucoup de dongles perturbent plus ou moins sérieusement le système, ou refusent de fonctionner sur certains PC imparfaitement compatibles, voire en présence de certaines imprimantes.

Le principe de la protection est le suivant : dès son lancement, le logiciel protégé interroge le dongle en envoyant un message codé sur la prise correspondante.

S'il est présent, le dongle répond par un autre message codé que le logiciel compare à ce qu'il attend. S'il y a coïncidence, l'exécution du programme se poursuit.

Si le dongle ne répond pas ou si le message reçu n'est pas conforme (utilisation d'un dongle différent ou tentative de *déplombage*), le logiciel réagit selon une procédure *défensive* : en général, il s'agit d'un simple retour au DOS avec ou sans message d'erreur, mais la riposte pourrait aller jusqu'à la destruction pure et simple du fichier exécutable !

Les dongles les plus simples ne contiennent guère que quelques connexions entre des broches peu utilisées en temps normal, tandis que les plus évolués sont équipés de circuits ASIC spécifiques et peuvent coûter plus de 1000 F à l'éditeur du logiciel, pesant donc lourdement sur le prix du produit...

Mais nous allons découvrir qu'il est possible de faire déjà très bien pour quelques dizaines de francs seulement, grâce à la participation aussi généreuse qu'involontaire de France Télécom !

Malgré son prix de revient d'environ 5 F (hors rémunération du distributeur et de l'inventeur), la télécarte épuisée peut être considérée comme de la matière première parfaitement gratuite.

Et pourtant, sa puce renferme une mémoire EPROM de 256 bits, à accès série, abritant notamment une zone inaltérable de 96 bits.

Il est plus que probable que chaque carte émise est unique au niveau de ce *numéro matricule*, puisque le nombre de combinaisons possibles est des milliers de fois supérieur à celui de toutes les télécartes que l'on peut raisonnablement espérer produire.

Quoi qu'il en soit, on peut considérer comme statistiquement négligeable la probabilité de découvrir deux cartes identiques.

Soigneusement découpée, la puce d'une carte peut donc être considérée comme un circuit intégré contenant une signature unique de 12 octets, facile à lire mais impossible à modifier.

Pourrait-on rêver mieux pour notre application, sachant que l'on peut facilement souder des fils de connexion sur la puce en question pour la relier au circuit utilisateur?

Evidemment, la technique de la carte à puce est brevetée, mais une telle application ne devrait pas soulever de problème puisque nous ne nous servons pas de la carte elle-même mais seulement du circuit intégré qu'elle contient.

Le passage de l'idée qui vient d'être exposée à une réalisation pratique n'est pas aussi simple qu'on pourrait le penser.

Il faut tout d'abord alimenter la puce sous à peu près 5 V, à partir des seules broches du port d'imprimante. Bien que les chiffres réels soient souvent assez inférieurs, la consommation de la puce est annoncée comme pouvant atteindre 60 mA.

Il faut ensuite appliquer des signaux bien précis aux broches de RAZ et d'horloge de la puce tout en interrogeant sa broche de sortie, pour lui faire *avouer* son contenu.

Enfin, des mesures efficaces doivent être prises pour éviter toute interférence entre ce dialogue PC-puce et les communications normales

PC.imprimante : pas question que l'imprimante aligne des caractères plus ou moins fantaisistes pendant l'interrogation du dongle, ou que celui-ci vienne perturber le bon déroulement d'opérations d'impression en réagissant intempestivement à certains codes !

Après diverses tentatives plus ou moins heureuses, nous avons retenu les principes suivants, qui se traduisent par le schéma de principe de la figure 4.35.

- Prélever la tension d'alimentation à l'aide de trois diodes sur les broches de STROBE, INIT, et AUTOFEED du PC, que le logiciel veillera à maintenir au niveau haut pendant le dialogue. Un avantage supplémentaire de ce choix est que tant que la ligne de STROBE est au niveau haut, l'imprimante ne tient aucun compte des codes pouvant circuler sur les lignes de données.

- Faire passer les signaux destinés aux broches de RAZ et d'horloge de la carte sur les lignes de données D0 et D1, la broche d'écriture de la puce étant maintenue à la masse puisque notre application n'utilise pas l'écriture dans la mémoire.

- Rapatrier les signaux présents en sortie de la puce vers le PC sur la ligne BUSY, ce qui implique de mettre en oeuvre une commutation entre le BUSY de l'imprimante et la sortie de la carte (nous sommes ici en présence de niveaux TTL et non trois états, ce qui interdit une simple mise en parallèle).

Un sélecteur de données CMOS 4512 convient parfaitement pour opérer cette commutation. Nous avons choisi de le commander par la ligne SLCT IN du PC : au niveau 0, celle-ci fait passer l'imprimante en ligne et simultanément aiguille le BUSY de l'imprimante vers le PC. Au niveau 1, par contre, elle met l'imprimante hors ligne et relie l'entrée BUSY du PC à la sortie de la puce.

A la seule exception de la ligne BUSY (broches 11 des DB 25), toutes les lignes du port parallèle traversent donc directement le dongle, quitte à recevoir quelques branchements en parallèle : l'organisation générale de ce port est donc aussi peu perturbée que possible.

Le plus commode est de graver une petite carte double face dont la figure 4.36 fournit le tracé de la face correspondant aux broches 14 à 25, et la figure 4.37 celui de la face correspondant aux broches 1 à 13, la ligne de la broche 11 étant naturellement coupée.

Il existe une place juste suffisante entre les deux rangées de broches des DB25 à souder pour introduire légèrement à force cette carte en époxy de l'épaisseur habituelle 16/10 mm. On exécutera ensuite 50 points de soudure en veillant à ne pas créer de court-circuit.

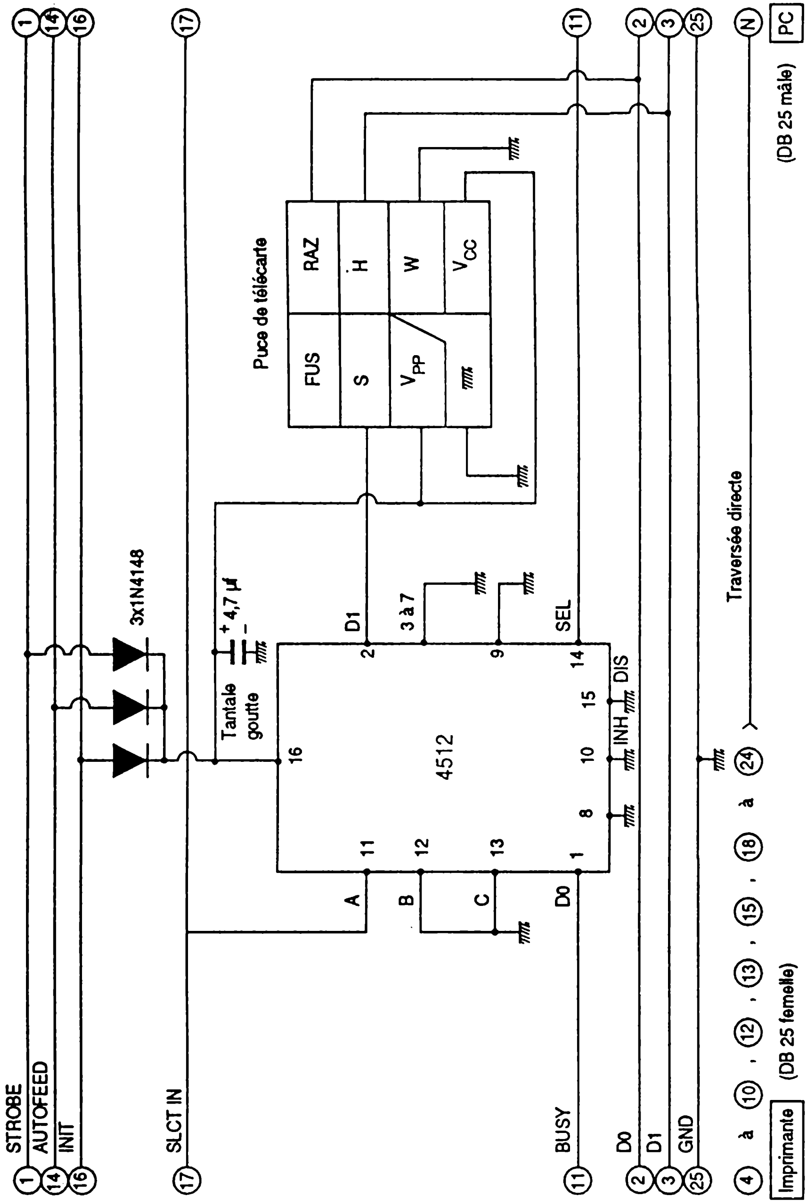


Fig. 4.35. – Schéma du dongle.

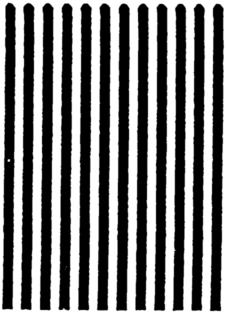


Fig. 4.36. – Circuit imprimé "bus" (envers).

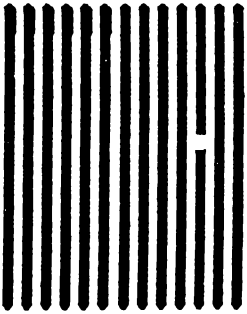


Fig. 4.37. – Circuit imprimé "bus" (endroit).

La partie électronique sera pour sa part câblée sur une petite carte simple face, dont la figure 4.38 reproduit le tracé.

Le câblage selon la figure 4.39 ne soulève de commentaire que pour le montage de la puce : celle-ci sera tout d'abord découpée aux ciseaux dans une télécarte quelconque, même incomplète (les demi-cartes ne sont pas rares !).

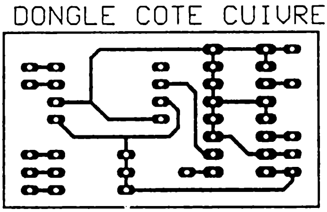


Fig. 4.38. – Circuit imprimé du dongle.

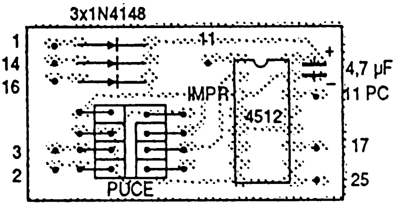


Fig. 4.39. – Plan de câblage du dongle.

Elle sera ensuite posée, avec la bonne orientation, sur le circuit imprimé et maintenue en place par rabattage des sept ou huit queues de résistances que l'on aura au préalable soudées perpendiculairement à la carte : le collage de la puce n'est pas nécessaire, mais pas interdit non plus.

Le soudage de ces fils sur la puce doit être aussi rapide que possible, un excès de chaleur pouvant vite décoller irrémédiablement ses contacts.

Une solution concurrente consiste à utiliser une colle conductrice comme l'ELECOLIT 340 ou le JELTARGENT.

Il ne reste plus, alors, qu'à raccorder sept fils en parallèle sur l'une ou l'autre des DB25 (1,2,3,14,16,17,25), et deux de part et d'autre de la coupure de la ligne des broches 11 : 11PC sur la broche 11 de la DB25 côté PC (mâle), et 11IMPR sur la broche 11 de la DB25 femelle (côté imprimante donc).

Cela fait, la carte peut être attachée ou collée contre le circuit *bus*, un carton isolant devant impérativement être placé entre les deux.

Le montage peut alors être enrobé dans une résine, ou simplement abrité dans un petit boîtier (par exemple un tronçon de HE222 HEILAND).

Chaque dongle ainsi construit étant en principe une pièce unique, il s'agit maintenant d'en relever la signature : d'abord pour le tester, et ensuite pour pouvoir incorporer cette information vitale dans le logiciel protégé.

Le petit programme BASIC de la figure 4.40 suffit pour procéder à cette lecture, le dongle étant installé entre la prise CENTRONICS du PC et une imprimante sous tension.

```
10 REM ----- DONGLE.BAS -----
20 E=889:S=888:U=890
30 OUT S,0:OUT U,4:CLS
40 OUT S,250:OUT S,248
50 PRINT"DATA ";
60 FOR G=1 TO 12
70 N=0
80 FOR H=7 TO 0 STEP -1
90 OUT S,249
100 D=INP(E):D= (D AND 128)
110 IF D<>128 THEN N=N+(2^H)
120 OUT S,251
130 NEXT H
140 PRINT N;",";:NEXT G
150 OUT 890,12
160 REM (c)1991 Patrick GUEULLE
```

Fig. 4.40. – Le logiciel de lecture du dongle.

Le résultat de l'exécution du programme doit être une ligne DATA ressemblant à celle de la figure 4.41.

Chaque valeur numérique est l'équivalent décimal de l'un des douze octets de la zone inaltérable de la puce, et doit donc être comprise entre 0 et 255.

Le second octet doit toujours être un 3, prouvant que la puce provient bien d'une télécarte et non d'une carte de cinéma ou de parcmètre.

```
DATA  195 , 3 , 98 , 56 , 187 , 54 , 37 , 42 , 183 , 255 , 16 , 6 ,
Ok
```

Fig. 4.41. – Exemple de résultat de la lecture du dongle.

Egalement, le dernier octet doit être un 6 si la puce provient d'une carte de 50 unités (cas le plus fréquent) ou un 19 si on l'a découpée sur une carte de 120 unités (éviter de sacrifier des 40 unités, qui ont tendance à prendre une valeur de collection non négligeable).

Si les douze octets sont tous à 0 ou à 255, alors il y a un problème : puce défectueuse ou endommagée au montage, erreur de câblage, mauvaise installation sur le PC, mauvais port d'imprimante (notre logiciel est écrit pour LPT1 :), ou tension d'alimentation insuffisante.

Dans ce dernier cas, on peut tenter d'ajouter quelques diodes pour prélever un surcroît d'énergie sur d'autres lignes, notamment les lignes de données D3 à D7 (tout dépend de la qualité de réalisation du port parallèle du PC, dont tous les dongles sont plus ou moins tributaires).

Il est maintenant bien évident que pour protéger un logiciel exécutable avec ce dongle, il suffit de lui ajouter une courte routine chargée de lire la mémoire de la puce, et de vérifier que son contenu est bien conforme à ce que l'on attend.

Le petit programme BASIC de la figure 4.42 se charge de cette opération, pourvu que sa ligne 20 contienne très exactement la liste de DATA élaborée par le programme DONGLE.BAS.

Sur cet exemple, la partie utile du logiciel se limite à la ligne 180, qui se borne à déclarer que le dongle est accepté. Dans la pratique, évidemment, on ajoutera les lignes 10 à 170 en tête du logiciel à protéger, quitte à en renuméroter les lignes auparavant (commande RENUM).

```
10 REM ----- PROTEGE.BAS -----
20 DATA 195,3,98,56,187,54,37,42,183,255,16,6
30 E=889:S=888:U=890
40 OUT S,0:OUT U,4:CLS
50 OUT S,250:OUT S,248
60 FOR G=1 TO 12
70 N=0:READ M
80 FOR H=7 TO 0 STEP -1
90 OUT S,249
100 D=INP(E):D= (D AND 128)
110 IF D<>128 THEN N=N+(2^H)
120 OUT S,251
130 NEXT H
140 IF N<>M THEN BEEP:PRINT"MAUVAIS DONGLE !":END
150 NEXT G
160 OUT 890,12
170 REM (c)1991 Patrick GUEULLE
180 PRINT"Dongle accepté!"
```

Fig. 4.42. – Exemple de programme protégé par le dongle.

Il est bien sûr hors de question de livrer le logiciel à son utilisateur sous cette forme : la lecture de son listing suffit pour comprendre instantanément qu'un simple RUN 180 permet de contourner la protection et donc de l'exécuter en l'absence du dongle !

Il est donc indispensable de compiler cette *source* BASIC en un *fichier exécutable* sous DOS, autrement dit en langage machine (fichier .EXE).

Cette opération est facile à exécuter avec un compilateur BASIC tel que TURBO-BASIC (Borland) ou QUICK-BASIC (Microsoft). Rappelons qu'un avantage supplémentaire de cette manoeuvre est que plus rien d'évident ne montre que le programme original a été écrit en BASIC : même un listing de quelques lignes se trouve converti en un impressionnant fichier de plusieurs dizaines de k-octets, parfaitement indéchiffrable.

En ce qui nous concerne, la protection se trouve noyée dans cette masse, et donc pratiquement impossible à isoler.

Bien entendu, le même principe serait applicable à tout autre langage compilé : le programme de la figure 4.42 est suffisamment simple pour que son algorithme puisse facilement être adapté en TURBO-PASCAL ou en C, par exemple.

Dans ce cas, on pourra avoir à ralentir artificiellement ces langages très rapides, afin de ne pas aller plus vite que la puce.

Bien que franchement expérimental, ce montage a été essayé avec succès sur divers types de compatibles PC. Cela ne veut pas dire, cependant, qu'il fonctionnera impeccablement sur n'importe quel système : peu de dongles en sont d'ailleurs capables...

En règle générale, il est nécessaire qu'une imprimante soit connectée, et mise sous tension : cela afin que les résistances de tirage qu'elle possède sur les lignes servant à l'alimentation de la puce fournissent du courant plutôt que d'en consommer. Mais il peut très bien arriver que le dongle fonctionne seul, ou relié à une imprimante hors tension.

Liste des composants du dongle :

Condensateur :
C1 : 4,7 µF 10 V tantale goutte.

Circuits intégrés :
CI1 : CD 4512B,
CI2 : puce de télécarte.

Autres semiconducteurs :
D1 à D3 : 1N 4148.

Divers :
1 fiche DB 25 mâle à souder,
1 fiche DB 25 femelle à souder.

⑤ APPLICATIONS DES PAL

Bien que les applications des réseaux logiques programmables PAL soient extrêmement nombreuses, ces composants ne sont pas encore tout à fait aussi populaires que les EPROM. C'est dommage, car nous allons constater qu'ils peuvent résoudre fort élégamment des problèmes relativement complexes, même si le recours à une EPROM demeure parfois une solution concurrente.

UN DOUBLE DÉ ÉLECTRONIQUE

La version électronique du jeu de dés est un grand classique des catalogues des fabricants de kits et des recueils de schémas de montages.

En règle générale, il faut au moins quatre ou cinq boîtiers CMOS ou TTL pour reconstituer l'équivalent d'un seul dé équipé de sept diodes LED.

L'utilisation d'un PAL apporte une amélioration décisive : un seul boîtier (PAL 16R8) suffit pour réaliser cette fois un double dé, tandis que la possibilité existe de rendre incopiable le schéma interne qu'on y a programmé !

Le schéma théorique de la figure 5.1 démontre à quel point l'utilisation de PAL, même de bas de gamme, simplifie les systèmes logiques : un seul circuit intégré PAL 16R8 et un multivibrateur à deux transistors suffisent pour réaliser ce montage au fonctionnement déjà relativement complexe.

Une programmation soigneusement étudiée permet de réaliser un compteur-décodeur parfaitement adapté à cet usage à partir des huit bascules et du réseau de portes que contient ce PAL très courant.

Moyennant quoi, la majeure partie du circuit imprimé dont la figure 5.2 donne le tracé, sert à assembler quatorze diodes LED selon une disposition rappelant à s'y méprendre deux dés juxtaposés.

Après câblage selon la figure 5.3 et raccordement à une pile plate de 4,5 V, chaque appui sur le bouton-poussoir doit déclencher un nouveau jet des dés, si toutefois le PAL 16R8 a été programmé correctement...

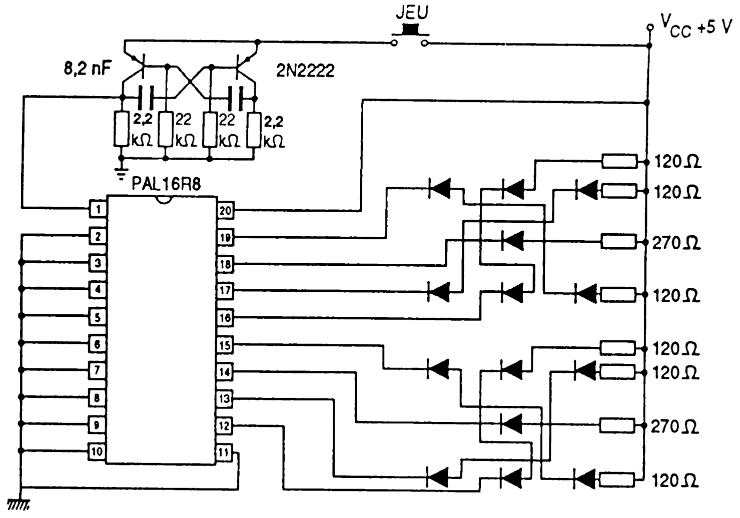


Fig. 5.1. – Schéma du double dé.

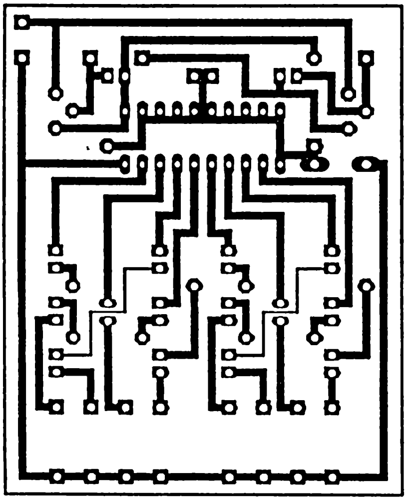


Fig. 5.2. – Circuit imprimé du double dé.

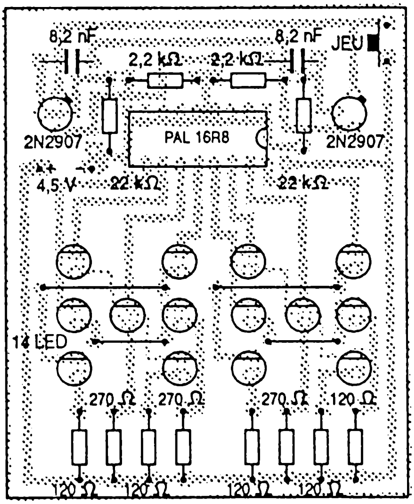


Fig. 5.3. – Plan de câblage du double dé.

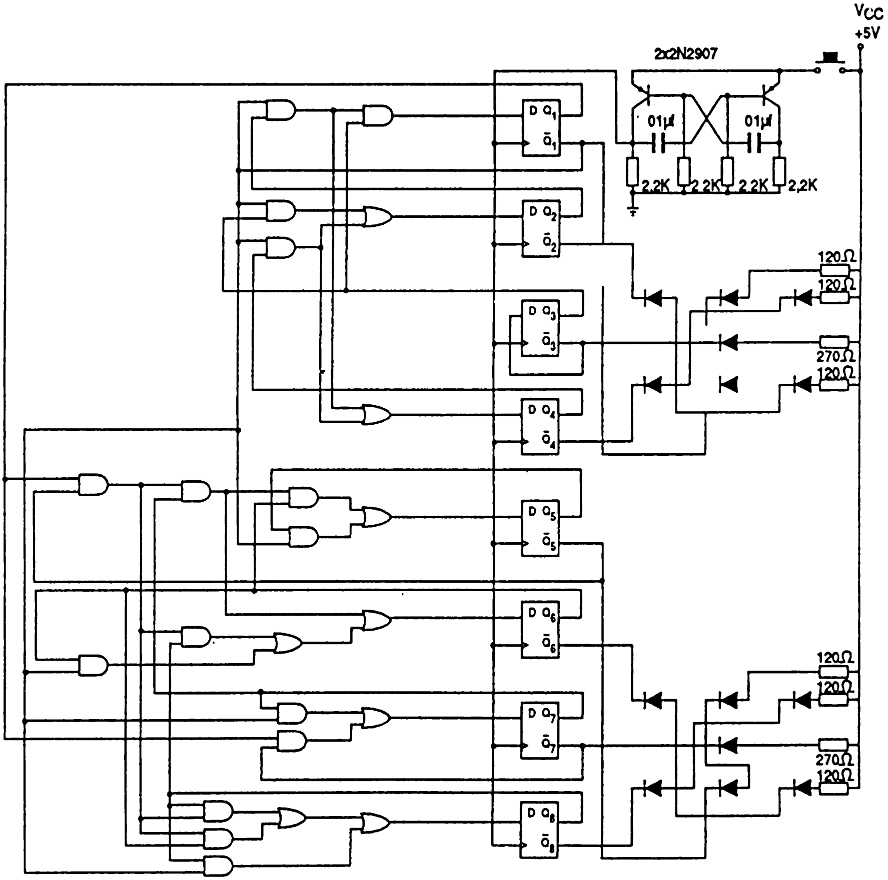


Fig. 5.4. – Schéma équivalent du double dé.

		0000	0000	0011	1111	1111	2222	2222	2233
term	pin	0123	4567	8901	2345	6789	0123	4567	8901
0	19	-X--	---X	----	--X-	----	----	----	----
1		-X--	----	---X	--X-	----	----	----	----
2		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
3		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
4		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
5		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
6		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
7		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
8	18	-X--	--X-	----	----	----	----	----	----
9		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
10		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
11		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
12		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
13		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
14		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
15		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
16	17	-X-X	----	----	--X-	----	----	----	----
17		-X--	----	---X	--X-	----	----	----	----
18		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
19		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
20		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
21		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
22		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
23		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
24	16	---X	---X	----	--X-	----	----	----	----
25		X---	----	----	----	----	----	----	----
26		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
27		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
28		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
29		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
30		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
31		XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX

Fig. 5.6. – Le plan des fusibles du double dé.

```
                  0000 0000 0011 1111 1111 2222 2222 2233
term    pin       0123 4567 8901 2345 6789 0123 4567 8901

32      15        -X-- ---- ---- ---X ---- ---X ---- --X-
33                -X-- ---- ---- ---X ---- ---- ---X --X-
34                -X-- ---- ---- --X- ---X ---- ---- ----
35                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
36                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
37                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
38                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
39                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

40      14        -X-- ---- ---- ---X ---- --X- ---- ----
41                -X-- ---- ---- --X- ---- ---X ---- ----
42                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
43                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
44                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
45                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
46                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
47                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX


48      13        -X-- ---- ---- ---X ---X ---- ---- --X-
49                -X-- ---- ---- ---X ---- ---- ---X --X-
50                -X-- ---- ---- --X- ---- ---- ---X ----
51                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
52                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
53                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
54                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
55                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

56      12        -X-- ---- ---- ---X ---X ---X ---- --X-
57                -X-- ---- ---- --X- ---- ---- ---- ---X
58                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
59                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
60                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
61                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
62                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
63                XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

L'objectif de cette programmation est de configurer le circuit interne du PAL selon le schéma de la figure 5.4, qui a été étudié d'une façon tout à fait classique et qui pourrait parfaitement être câblé à partir de composants standards : 8 bascules, 23 portes, et 2 transistors.

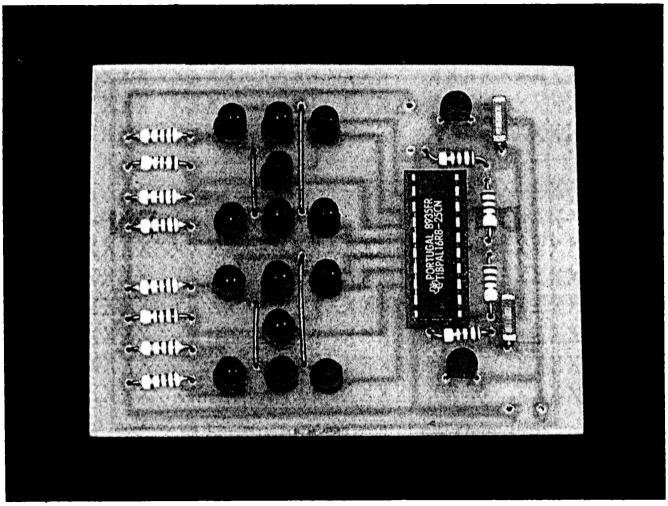
Pour transférer ce schéma dans un PAL, on pourrait bien sûr opérer manuellement sur un plan des fusibles, mais il est plus confortable de repartir des équations qui ont servi de base à son élaboration par la méthode des tableaux de Karnaugh.

Il suffit de soumettre ces équations à un compilateur logique (en l'occurence PALASM) pour que ce logiciel établisse automatiquement un plan des fusibles et même le fichier JEDEC destiné au programmateur de PAL.

Ce plan des fusibles, reproduit à la figure 5.6, décrit entièrement la programmation du PAL 16R8, dont la figure 5.7 rappelle le schéma interne.

$$D_1 = \overline{Q_1}\,Q_2\,Q_3$$
$$D_2 = \overline{Q_1}\,Q_3 + \overline{Q_1}\,Q_4$$
$$D_3 = \overline{Q_3}$$
$$D_4 = \overline{Q_1}\,Q_2 + \overline{Q_1}\,Q_4$$
$$D_5 = \overline{Q_1}\,Q_5 + Q_1\,\overline{Q_5}\,Q_6\,Q_7$$
$$D_6 = \overline{Q_1}\,Q_6 + Q_1\,\overline{Q_5}\,Q_7 + Q_1\,\overline{Q_5}\,Q_8$$
$$D_7 = \overline{Q_1}\,Q_7 + Q_1\,\overline{Q_7}$$
$$D_8 = \overline{Q_1}\,Q_8 + Q_1\,\overline{Q_5}\,Q_6 + Q_1\,\overline{Q_5}\,Q_8$$

Fig. 5.5. – Les équations du double dé.



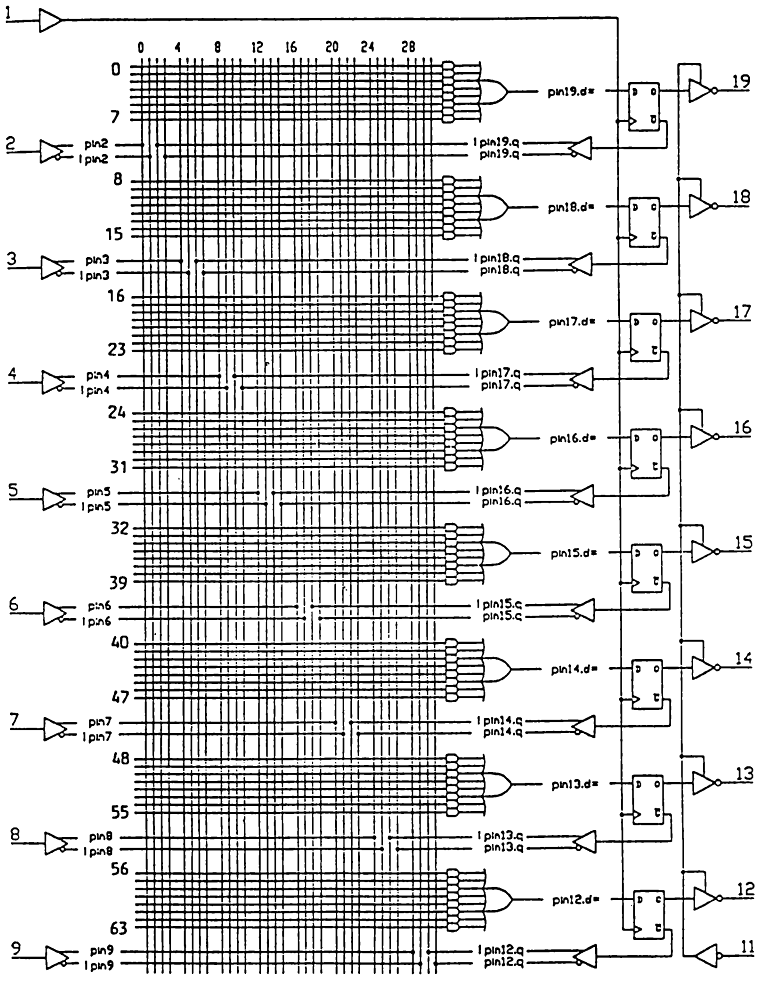


Fig. 5.7. – Schéma interne du PAL 16R8.

```
Written to disk by the SPRINTplus:  Logic programmer
for the device Cypress PALC16R8*
F0*
QF2048*
L0 10111110111111011111111111111111*
L32 10111111111011011111111111111111*
L256 10111101111111111111111111111111*
L512 10101111111111011111111111111111*
L544 10111111111011011111111111111111*
L768 11101110111111011111111111111111*
L800 01111111111111111111111111111111*
L1024 10111111111111101111111011111101*
L1056 10111111111111101111111111101101*
L1088 10111111111111011110111111111111*
L1280 10111111111111101111110111111111*
L1312 10111111111111011111111011111111*
L1536 10111111111111101110111111111101*
L1568 10111111111111101111111111101101*
L1600 10111111111111011111111111101111*
L1792 10111111111111101110111011111101*
L1824 10111111111111011111111111111110*
L2016 00000000000000000000000000000000*
C394D*
♥0000
```

Fig. 5.8. – Fichier JEDEC du double dé.

Un X correspond à un fusible intact, tandis qu'un tiret - indique un fusible détruit.

Sur 64 lignes de 32 fusibles ou termes de produits, 47 sont laissées intactes, ce qui montre que cette application pourtant déjà performante n'utilise pas à plein les ressources du 16R8, PAL réputé simple !

Le fichier JEDEC de la figure 5.8 ne reprend pas, à part la dernière, les lignes ne comportant que des fusibles intacts, sur lesquelles le programmateur n'a en effet rien à faire.

Chaque 0 d'une ligne JEDEC correspond à un X du plan de fusibles (fusible intact), tandis que chaque 1 JEDEC correspond à un fusible détruit (tiret du plan).

Chaque ligne JEDEC commence par un L majuscule immédiatement suivi (sans espace) du numéro du premier fusible de cette même ligne. Suit un espace, la description des 32 fusibles de la ligne, puis une étoile clôturant la ligne.

Les numéros des fusibles commencent à zéro pour le premier fusible de la première ligne, et vont jusqu'à 2047 pour le trente deuxième fusible de la soixante quatrième ligne (numérotée 63 puisque la première ligne est numérotée zéro tout comme le premier fusible de la première ligne).

Ainsi s'explique l'indication QF2048* (Quantité de Fusibles) qui rappelle tout simplement que le PAL 16R8 contient 2048 fusibles.

Cette indication est facultative, comme le sont d'ailleurs toutes les lignes JEDEC ne commençant pas par un L majuscule et ne décrivant donc pas une ligne de fusibles (par exemple la clef de contrôle C394D*).

C'est ce fichier que nos lecteurs désireux de réaliser le montage devront saisir, à l'aide d'un simple éditeur de textes, avant de le soumettre à un programmateur approprié (par exemple le programmateur pour les PAL CMOS de marque CYPRESS décrit dans notre ouvrage *COMPOSANTS ÉLECTRONIQUES PROGRAMMABLES*).

Il importe de prendre conscience du fait que, tout comme dans un programme en langage machine, une seule erreur dans le fichier JEDEC peut rendre le PAL programmé complètement inutilisable.

Il est d'usage de donner l'extension .JED aux fichiers JEDEC enregistrés sur des disquettes de PC. Dans notre cas, le fichier du double dé se nomme DICE.JED.

Liste des composants du double dé :

Résistances (5% 1/4 W)
R1 à R6 : 120 Ω,
R7 et R8 : 270 Ω,
R9 et R10 : 2,2 kΩ,
R11 et R12 : 22 kΩ.

Condensateurs :
C1 et C2 : 8,2 nF.

Transistors :
T1 et T2 : 2N 2222 ou BC 107.

Circuits intégrés :
CI1 : PALC 16R8 CYPRESS,
ou autre PAL 16R8 (selon programmateur disponible).

Autres semiconducteurs :
D1 à D14 : LED rouges.

Divers :
1 bouton-poussoir.
1 pile plate 4,5 V ou alim 5 V.

UN DÉCODEUR 7 SEGMENTS PERSONNALISÉ

Notre propos ne sera pas ici de décrire véritablement un montage, mais plutôt de montrer comment la technique des composants programmables permet de fabriquer un circuit intégré introuvable dans le commerce, et pourtant nécessaire pour mener à bien une réalisation particulière.

Nous ne fournirons donc pas de tracé de circuit imprimé, la mise en oeuvre pratique du composant spécifique ainsi créé dépendant des besoins exacts de chacun de nos lecteurs.

Le problème posé consiste à réaliser un circuit intégré décodeur 7 segments et mémoire, spécialement adapté au récepteur de numérotation téléphonique DTMF SSI 202 de Silicon Systems.

La figure 5.9 montre que le code hexadécimal utilisé diffère de celui que reconnaissent les décodeurs sept segments habituels, notamment au niveau du zéro.

	Hexadecimal			
Digit	D8	D4	D2	D1
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
0	1	0	1	0
•	1	0	1	1
#	1	1	0	0
A	1	1	0	1
B	1	1	1	0
C	1	1	1	1
D	0	0	0	0

Fig. 5.9. – Le code hexadécimal du SSI 202.

```
title ( décodeur 7 segments pour SSI 202 )
include p16r8;
define aa = pin19.d;
define bb = pin18.d;
define cc = pin17.d;
define dd = pin16.d;
define ee = pin15.d;
define ff = pin14.d;
define gg = pin13.d;
define AA = pin2;
define BB = pin3;
define CC = pin4;
define DD = pin5;
truth_table (
DD   CC   BB   AA   :  aa   bb   cc   dd   ee   ff   gg;
 0    0    0    0   :  0    1    1    1    1    0    1;
 0    0    0    1   :  0    1    1    0    0    0    0;
 0    0    1    0   :  1    1    0    1    1    0    1;
 0    0    1    1   :  1    1    1    1    0    0    1;
 0    1    0    0   :  0    1    1    0    0    1    1;
 0    1    0    1   :  1    0    1    1    0    1    1;
 0    1    1    0   :  1    0    1    1    1    1    1;
 0    1    1    1   :  1    1    1    0    0    0    0;
 1    0    0    0   :  1    1    1    1    1    1    1;
 1    0    0    1   :  1    1    1    1    0    1    1;
 1    0    1    0   :  1    1    1    1    1    1    0;
 1    0    1    1   :  1    0    0    1    0    0    1;
 1    1    0    0   :  0    0    1    1    1    0    1;
 1    1    0    1   :  1    1    1    0    1    1    1;
 1    1    1    0   :  0    0    1    1    1    1    1;
 1    1    1    1   :  1    0    0    1    1    1    0;
)
```

Fig. 5.10. – La table de vérité du décodeur à PAL.

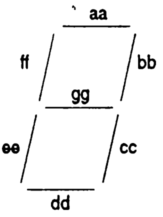


Fig. 5.11. – Affectation des 7 segments.

Là encore, le PAL 16R8 convient très bien, ses huit bascules pouvant servir de verrous pour la mémorisation du chiffre ou signe à afficher.

Grâce au compilateur logique (cette fois PROLOGIC), il suffit de rédiger la table de vérité de la figure 5.10 (correspondant à l'affectation des segments de la figure 5.11) pour que soit automatiquement généré le fichier JEDEC de la figure 5.12.

Bien entendu, ce fichier est directement exploitable par le programmateur de PAL, pour transformer un simple 16R8 en circuit intégré spécifique.

```
proLogic Compiler
Texas Instruments V1.97
Copyright (C) 1989 INLAB, Inc.

        décodeur 7 segments pour SSI 202

p16r8 revision 89.2.11

*N_csidp16r8
*QP20
*QF2048
*F0
*L0000 11111111101101111111111111111111
*L0032 01111111011111111111111111111111
*L0064 11110111111110111111111111111111
*L0256 10111011111110111111111111111111
*L0288 01110111111110111111111111111111
*L0320 01111011111101111111111111111111
*L0352 10111111101111111111111111111111
*L0384 11111111101110111111111111111111
*L0512 10111111111101111111111111111111
*L0544 11111111011110111111111111111111
*L0576 01111111111110111111111111111111
*L0608 11111011111111111111111111111111
*L0768 01111011011110111111111111111111
*L0800 11110111111101111111111111111111
*L0832 10111111111101111111111111111111
*L0864 11111111101101111111111111111111
*L0896 10110111111111111111111111111111
*L0928 11110111101111111111111111111111
*L0960 10111111101111111111111111111111
*L1024 11111111011101111111111111111111
*L1056 10110111111111111111111111111111
*L1088 10111111101111111111111111111111
*L1280 10111111011110111111111111111111
*L1312 11111011101101111111111111111111
*L1344 0111ı(11011111111111111111111111
*L1376 10110111111101111111111111111111
*L1408 11110111011101111111111111111111
*L1536 01110111101111111111111111111111
*L1568 11111011111101111111111111111111
*L1600 10111111011111111111111111111111
*L1632 11111011011111111111111111111111
*L1664 10111111111110111111111111111111
*C7BE6
*♥26D6
```

Fig. 5.12. – Le fichier JEDEC du décodeur à PAL.

En plus du fichier JEDEC, le logiciel PROLOGIC fournit aussi toute une documentation que nous reproduisons ici :

- Les équations simplifiées découlant de la table de vérité (figure 5.13).

- Le plan des fusibles (figure 5.14).

Naturellement, la même démarche serait applicable à la fabrication d'un circuit décodeur adapté à n'importe quel code.

```
proLogic Compiler
Texas Instruments V1.97
Copyright (C) 1989 INLAB, Inc.
Signal Specifications


pin13.d=
          !pin4 & pin3 & pin2
        # pin5 & !pin3
        # pin4 & !pin2
        # pin4 & !pin3
        # !pin5 & !pin2

pin14.d=
          !pin5 & pin4 & !pin2
        # pin5 & !pin4 & !pin3
        # pin4 & !pin3 & pin2
        # pin5 & pin3 & !pin2
        # pin5 & pin4 & pin3

pin15.d=
          pin5 & pin4
        # pin3 & !pin2
        # !pin4 & !pin2

pin16.d=
          !pin5 & pin4 & !pin3 & pin2
        # pin5 & pin3
        # pin5 & !pin2
        # pin5 & !pin4
        # pin3 & !pin2
        # !pin4 & pin3
        # !pin4 & !pin2

pin17.d=
          pin5 & !pin2
        # !pin5 & pin4
        # !pin5 & pin2
        # !pin3

pin18.d=
          !pin5 & !pin3 & !pin2
        # !pin5 & pin3 & pin2
        # pin5 & !pin3 & pin2
        # !pin4 & !pin2
        # !pin5 & !pin4

pin19.d=
          pin5 & !pin4
        # pin4 & pin2
        # !pin5 & pin3
```

Fig. 5.13. – Les équations du décodeur.

```
proLogic Compiler
Texas Instruments V1.97
Copyright (C) 1989 INLAB, Inc.
Fuse Plot

        décodeur 7 segments pour SSI 202

p16r8 revision 89.2.11

                 11 1111 1111 2222 2222 2233
    0123 4567 8901 2345 6789 0123 4567 8901

  0 ---- ---- -X-- X--- ---- ---- ---- ----  +
  1 X--- ---- X--- ---- ---- ---- ---- ----  +
  2 ---- X--- ---- -X-- ---- ---- ---- ----  +
  3 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +   pin19.d
  4 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
  5 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
  6 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
  7 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +

  8 -X-- -X-- ---- -X-- ---- ---- ---- ----  +
  9 X--- X--- ---- -X-- ---- ---- ---- ----  +
 10 X--- -X-- ---- X--- ---- ---- ---- ----  +
 11 -X-- ---- -X-- ---- ---- ---- ---- ----  +   pin18.d
 12 ---- ---- -X-- -X-- ---- ---- ---- ----  +
 13 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
 14 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
 15 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +

 16 -X-- ---- ---- X--- ---- ---- ---- ----  +
 17 ---- ---- X--- -X-- ---- ---- ---- ----  +
 18 X--- ---- ---- -X-- ---- ---- ---- ----  +
 19 ---- -X-- ---- ---- ---- ---- ---- ----  +   pin17.d
 20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
 21 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
 22 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
 23 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +

 24 X--- -X-- X--- -X-- ---- ---- ---- ----  +
 25 ---- X--- ---- X--- ---- ---- ---- ----  +
 26 -X-- ---- ---- X--- ---- ---- ---- ----  +
 27 ---- ---- -X-- X--- ---- ---- ---- ----  +   pin16.d
 28 -X-- X--- ---- ---- ---- ---- ---- ----  +
 29 ---- X--- -X-- ---- ---- ---- ---- ----  +
 30 -X-- ---- -X-- ---- ---- ---- ---- ----  +
 31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX  +
```

Fig. 5.14. – Le plan des fusibles du décodeur.

```
32 ---- ---- X--- X--- ---- ---- ---- ----   +
33 -X-- X--- ---- ---- ---- ---- ---- ----   +
34 -X-- ---- -X-- ---- ---- ---- ---- ----   +
35 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +   pin15.d
36 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
37 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
38 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
39 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +

40 -X-- ---- X--- -X-- ---- ---- ---- ----   +
41 ---- -X-- -X-- X--- ---- ---- ---- ----   +
42 X--- -X-- X--- ---- ---- ---- ---- ----   +
43 -X-- X--- ---- X--- ---- ---- ---- ----   +   pin14.d
44 ---- X--- X--- X--- ---- ---- ---- ----   +
45 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
46 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
47 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +

48 X--- X--- -X-- ---- ---- ---- ---- ----   +
49 ---- -X-- ---- X--- ---- ---- ---- ----   +
50 -X-- ---- X--- ---- ---- ---- ---- ----   +
51 ---- -X-- X--- ---- ---- ---- ---- ----   +   pin13.d
52 -X-- ---- ---- -X-- ---- ---- ---- ----   +
53 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
54 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
55 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +

56 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
57 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
58 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
59 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +   pin12.d
60 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
61 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
62 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +
63 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX   +

   : :  : :  : :  : :  : :  : :  : :  : :
pin2 :  3 :  4 :  5 :  6 :  7 :  8 :  9 :
     :    :    :    :    :    :    :    :
   pin1   18   17   16   15   14   13   11


Legend:

  X  : Cell intact     (JEDEC 0)
  -  : Cell programmed (JEDEC 1)

  X- : True input term, Complement register term
  -X : Complement input term, True register term
  XX : Any XX pair in a product term yields product term LOW.
  -- : No input term (don't care).  A product term comprised
       entirely of -- yields product term HIGH.
```

ANNEXE

Saisi à l'aide d'un éditeur de texte ou mieux d'un "scanner" associé à un logiciel de reconnaissance de caractères (OCR), ce fichier de valeurs décimales peut être directement enregistré dans une EPROM (une moitié de 27C64) en vue de sa diffusion par le reproducteur autonome.

Le montage fera alors office de "sirène parlante" criant "AU VOLEUR!"

255	255	253	253	251	251	249	249	247	247	245	245	243
243	241	241	239	238	237	236	235	234	233	232	231	230
229	228	227	226	225	224	223	222	221	220	219	218	217
216	215	214	213	212	211	210	209	208	207	206	205	204
203	202	201	200	199	198	197	196	195	194	193	192	191
190	189	188	187	186	185	184	183	182	181	180	179	178
177	176	175	174	173	172	171	170	169	168	167	166	165
164	163	162	161	160	159	158	157	156	155	154	153	152
151	150	149	148	147	146	145	144	143	142	141	140	139
138	137	136	135	134	133	132	131	130	129	128	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	127	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
127	126	126	126	126	126	126	126	126	126	126	125	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	127	126	126	126	126	126	126	126	126	125	126	127
127	126	126	127	125	126	126	125	126	127	124	129	128
129	124	126	123	120	126	125	127	127	127	127	124	128
128	128	129	124	126	122	124	126	127	127	127	126	126

127	127	127	127	126	124	124	124	125	124	127	127	127
127	127	126	125	126	127	122	127	125	131	124	130	127
122	123	122	123	124	125	124	131	126	127	124	129	130
128	130	126	122	125	121	124	126	127	126	127	127	124
131	125	131	125	124	124	124	124	126	127	127	126	129
123	126	123	126	127	135	129	136	128	125	122	118	123
123	125	126	131	127	127	126	129	133	134	130	124	121
117	119	120	125	124	131	124	131	127	124	131	131	124
128	127	120	124	124	124	120	129	116	129	124	137	132
141	128	128	121	117	119	116	121	124	131	124	131	125
131	134	132	132	130	121	120	117	122	119	124	127	127
126	129	130	131	129	130	130	120	124	117	123	123	134
130	136	128	130	122	123	123	123	127	122	125	122	123
121	131	131	134	132	128	125	121	119	122	123	126	126
125	126	127	127	126	129	131	130	130	120	121	115	123
122	133	128	136	191	127	120	127	126	129	124	126	123
122	123	127	127	134	131	128	128	120	123	121	127	124
130	124	124	124	130	126	124	127	122	126	128	133	134
133	128	126	123	116	123	121	122	125	127	127	126	130
129	133	133	128	129	120	121	122	121	126	127	125	127
126	125	128	134	130	124	120	121	121	124	129	130	134
130	124	124	120	121	126	127	127	126	125	127	127	124
128	128	129	124	126	121	127	127	126	124	128	129	127
122	126	122	124	130	128	128	132	130	127	121	127	123
125	127	124	125	124	125	124	131	124	128	124	125	125
127	126	131	127	127	126	127	125	126	124	126	126	129
128	128	130	126	125	124	127	127	124	131	126	125	125
127	126	126	125	124	125	126	126	127	127	126	127	127
126	126	126	124	125	126	127	121	125	125	129	125	128
131	126	126	124	127	124	131	125	125	127	123	127	127
124	124	128	127	124	129	127	126	125	127	127	123	124
131	126	125	128	131	126	125	126	127	127	127	124	129
127	126	126	125	124	128	125	125	126	126	125	130	124
128	129	124	128	130	124	124	124	127	126	127	126	131
127	125	126	126	126	127	124	130	127	125	125	125	126
127	123	127	126	124	131	124	130	126	129	126	126	126
123	124	125	126	127	124	131	126	130	125	127	126	125
124	129	127	127	126	127	124	131	122	124	124	127	124
131	126	126	126	122	124	131	124	131	124	131	126	124
124	131	124	128	124	131	123	127	124	130	126	127	123
127	120	131	127	126	127	124	131	121	128	129	123	124
120	128	128	128	127	124	130	124	131	124	131	124	131
124	130	124	131	126	127	119	127	126	125	129	131	127
126	124	131	126	135	126	129	124	125	119	128	124	129
127	125	120	132	131	127	123	127	119	124	131	127	127
124	128	124	124	123	127	131	124	127	124	129	125	128
125	120	124	126	124	128	124	127	120	124	125	131	127
126	124	128	113	127	127	127	134	132	128	131	124	131
124	132	118	117	120	121	126	136	128	128	128	127	126
139	124	107	108	121	119	143	140	139	134	129	116	136
136	123	108	113	103	127	136	139	124	131	123	125	140
140	111	95	109	99	132	152	144	132	137	123	124	136
128	103	107	95	127	142	136	128	127	112	143	140	131

116	121	107	118	140	131	124	133	119	124	144	131	120
124	107	111	124	131	127	140	131	124	136	128	128	112
83	95	95	135	146	179	131	156	136	148	124	116	71
87	87	127	132	160	144	157	148	147	94	108	81	99
124	155	134	158	136	147	140	137	106	107	87	103	118
139	128	145	132	139	140	144	128	96	115	107	124	152
147	127	140	131	137	128	107	87	111	118	136	140	129
117	129	131	140	128	83	116	131	134	160	145	103	132
136	132	126	113	79	119	139	140	148	147	115	135	138
128	75	92	103	127	184	167	141	156	156	143	132	104
67	79	107	123	148	155	127	156	168	152	96	88	99
95	156	147	118	147	148	159	144	129	71	95	115	124
140	129	111	126	148	136	136	99	71	124	153	132	140
129	127	156	136	112	99	91	103	140	147	127	135	143
143	140	128	35	94	123	120	184	131	159	188	168	128
116	65	62	128	104	123	126	155	156	184	131	63	124
115	124	144	99	111	156	160	128	124	99	92	156	131
118	127	123	125	140	129	105	111	119	144	152	131	125
132	136	140	120	97	95	111	127	133	141	145	145	144
144	99	47	124	113	108	160	107	156	192	160	136	112
112	99	110	107	79	126	150	140	109	95	124	131	140
160	107	124	160	136	120	99	89	126	129	119	118	138
146	148	96	104	136	131	142	131	111	158	162	128	113
107	112	120	120	103	103	125	143	134	136	103	95	156
163	99	124	131	116	172	131	95	126	123	124	131	115
135	135	140	129	67	92	152	115	148	131	119	168	163
108	127	119	124	130	114	125	131	132	129	83	116	134
127	124	131	111	142	152	130	134	129	126	132	131	109
123	116	128	103	95	124	128	128	152	127	127	156	135
126	131	115	132	132	113	124	119	116	120	99	108	136
131	124	135	119	140	160	129	128	129	124	140	129	110
113	115	120	128	105	103	126	148	129	126	135	119	156
160	107	124	129	116	140	115	111	131	119	124	115	79
124	131	108	160	131	124	192	131	126	137	103	148	131
95	124	128	112	97	96	127	127	124	131	115	156	161
129	136	131	116	148	131	110	124	120	120	99	95	127
119	124	139	111	164	147	128	140	113	124	146	107	108
108	108	128	112	83	110	152	131	140	163	119	172	161
99	126	131	108	136	99	95	124	140	129	67	124	147
108	204	131	92	208	131	126	131	63	124	129	99	126
160	128	67	132	131	124	224	131	124	195	128	152	115
79	94	128	99	110	158	128	94	124	147	111	172	131
127	164	144	128	96	96	92	136	107	95	127	143	156
147	75	124	163	118	154	131	95	172	147	91	103	99
128	160	131	111	156	152	119	52	131	111	188	163	102
220	163	132	112	51	92	156	105	127	174	131	36	160
131	92	227	95	124	202	135	95	125	67	78	129	99
126	176	129	32	160	131	127	191	111	127	191	135	111
96	67	88	144	96	116	155	156	184	144	65	124	160
99	140	131	95	188	163	79	110	123	124	152	131	111
168	160	67	39	126	111	156	195	95	188	200	128	96
67	31	126	129	111	200	192	67	148	131	63	188	143
92	184	99	108	128	83	126	135	95	124	176	129	20

128	131	124	226	127	124	209	131	92	89	47	108	145
83	110	172	180	184	160	51	63	142	99	132	131	127
204	192	99	95	99	119	144	131	124	160	129	31	92
99	120	208	147	159	192	144	96	96	35	63	127	135
188	194	47	124	128	129	128	162	123	188	193	103	88
81	75	124	131	111	188	193	71	46	88	112	184	194
127	124	180	129	96	67	43	148	131	135	176	168	148
75	39	95	127	156	161	131	188	192	128	96	67	63
124	129	123	158	176	176	128	35	60	131	127	188	131
126	192	160	98	75	55	95	164	163	172	176	131	14
127	79	124	168	179	188	224	131	84	48	35	63	159
140	216	192	64	127	107	83	108	195	112	240	192	131
60	67	31	124	139	143	188	193	47	31	78	107	124
227	127	188	192	128	96	67	31	126	173	139	172	180
81	47	67	79	150	192	161	188	192	128	96	65	39
95	159	143	188	184	144	35	46	95	95	156	194	156
220	160	104	73	35	55	118	168	168	204	176	131	3
62	83	124	220	160	184	224	160	96	67	15	63	149
163	156	204	176	128	32	95	63	108	225	145	156	196
128	96	59	15	60	168	177	180	192	160	67	28	99
79	124	211	150	220	192	128	60	49	31	95	167	175
204	192	99	15	63	79	126	189	159	190	192	131	60
35	31	111	159	157	192	192	144	67	11	63	111	140
224	187	188	176	128	72	35	39	95	173	180	190	186
148	128	33	47	95	127	236	193	156	192	144	86	43
23	63	127	190	226	208	160	67	28	83	79	127	173
179	228	224	128	80	19	7	63	127	156	252	208	131
28	65	48	104	160	151	188	248	160	128	64	3	31
111	143	190	240	192	96	93	63	63	124	129	158	228
160	128	81	1	63	127	159	172	224	192	131	37	35
63	124	176	163	220	208	160	96	67	3	63	127	175
188	208	160	99	30	63	63	119	165	179	196	200	144
96	65	15	63	125	171	218	224	176	131	7	47	59
111	156	177	208	224	176	96	65	3	31	95	159	188
240	200	160	66	7	47	79	126	191	190	228	200	144
68	19	3	63	127	175	228	224	192	129	3	31	63
108	179	187	220	192	192	80	19	3	63	127	175	252
240	192	131	15	23	63	102	145	179	228	240	192	128
64	3	31	95	143	223	252	208	144	64	3	31	63
127	159	223	234	208	144	72	3	7	63	119	157	236
240	192	136	64	1	31	95	127	191	220	224	192	144
64	11	15	63	127	191	220	224	192	144	64	3	31
95	127	195	222	224	192	144	64	17	7	63	125	179
220	240	192	144	64	3	31	79	125	177	223	220	192
128	64	3	15	79	127	191	252	240	192	128	32	3
47	95	127	191	222	244	200	136	64	1	7	63	127
175	236	248	208	144	96	3	15	61	125	175	220	252
224	160	96	16	3	31	95	151	207	252	232	176	128
80	3	15	79	127	191	254	240	192	144	96	11	11
63	111	175	204	248	224	160	128	80	3	15	63	127
191	252	232	176	160	96	19	3	55	111	159	223	232
224	160	128	64	3	31	95	127	223	252	208	176	144
64	3	15	63	127	191	236	236	208	160	112	80	3

7	63	127	187	252	248	192	160	104	48	3	47	95
159	238	252	220	192	128	88	64	3	31	95	143	223
252	227	156	136	64	3	15	79	127	188	252	232	196
160	99	68	73	3	47	111	159	239	254	227	132	132
65	3	31	87	127	188	251	214	192	160	97	64	72
67	47	111	151	191	252	208	163	108	104	11	55	95
127	188	243	192	176	128	83	57	84	95	63	103	171
188	240	192	163	84	112	67	63	79	123	159	204	200
160	144	128	90	87	95	125	67	77	159	156	208	192
163	95	124	99	79	75	111	135	188	180	156	144	132
112	107	99	122	128	71	95	143	191	190	184	131	124
129	86	87	79	103	126	180	161	151	144	144	128	105
93	115	140	75	63	127	159	194	192	163	99	124	96
83	71	91	120	200	163	155	140	131	104	105	83	95
158	129	67	124	144	181	168	152	99	124	131	100	107
98	115	126	176	132	136	136	131	120	121	99	111	148
160	81	92	131	159	188	184	129	108	147	98	110	87
95	111	172	160	141	136	136	123	126	103	91	127	156
152	83	94	127	143	182	160	131	92	148	113	115	79
95	119	156	162	128	131	132	132	128	113	99	111	140
156	152	83	60	139	142	172	144	135	92	140	116	97
79	111	124	176	140	131	124	140	136	128	102	103	124
149	150	146	115	63	126	144	161	150	142	123	140	144
96	99	91	127	136	168	131	127	135	140	128	120	103
111	124	133	128	140	144	136	83	104	131	124	171	146
147	127	136	103	108	99	95	127	156	153	132	136	128
131	124	115	111	123	125	135	135	129	135	136	131	71
124	131	154	168	160	129	124	136	98	102	87	111	127
168	147	148	130	130	124	115	95	111	127	135	140	139
131	137	145	136	99	63	125	131	156	177	136	129	132
133	98	96	83	95	143	164	163	140	139	116	124	112
97	103	127	139	146	144	131	124	130	121	120	120	124
91	108	155	142	176	162	138	119	120	99	86	115	103
125	156	169	136	144	130	113	111	111	107	127	147	148
144	129	113	112	117	117	121	128	143	146	144	99	102
135	127	148	144	123	123	126	129	112	115	107	126	144
146	131	124	113	124	130	119	119	124	138	140	136	122
117	127	126	128	130	127	127	132	134	127	127	124	107
95	124	129	140	168	135	124	128	114	107	108	119	110
148	140	148	136	128	107	125	111	116	128	140	133	136
128	116	120	120	117	127	127	127	135	136	138	130	124
123	123	124	107	87	124	131	150	176	144	123	108	115
99	118	115	111	143	150	153	136	128	103	118	119	115
119	135	140	158	136	116	105	103	111	124	138	139	143
136	132	130	112	109	111	119	125	139	137	136	128	115
116	130	123	124	137	117	126	132	121	124	121	113	124
136	135	134	139	118	124	120	123	116	132	129	136	136
129	122	116	119	123	127	130	132	138	129	124	125	123
119	122	127	126	133	129	129	129	127	124	115	99	124
131	138	166	145	124	124	111	113	121	117	115	124	143
140	148	128	113	112	115	112	129	128	136	137	129	126
123	113	117	127	127	132	136	132	131	120	120	121	123
119	125	135	132	132	129	122	123	127	127	128	132	115

110	132	131	140	152	131	122	117	107	123	126	127	124
136	131	136	136	128	116	118	115	126	132	131	125	129
130	128	128	120	118	119	123	125	138	134	132	130	121
121	121	123	123	126	130	134	134	128	128	120	119	119
127	128	133	132	132	128	119	110	123	111	136	146	136
187	122	115	125	119	116	131	126	133	140	139	124	128
112	117	127	126	130	132	131	127	124	127	120	124	123
123	125	134	132	136	131	122	123	119	123	127	125	128
133	135	128	131	120	121	123	123	125	131	131	130	128
129	125	125	127	126	115	110	125	122	148	145	140	128
120	115	110	123	116	135	130	131	134	138	131	124	121
115	115	123	131	136	136	132	124	126	123	120	125	125
127	127	127	130	128	131	121	123	123	126	130	130	130
126	126	126	126	125	125	125	126	127	125	128	130	124
124	125	127	126	127	124	129	127	126	126	127	126	124
116	126	123	124	137	136	134	124	123	112	123	122	127
124	131	127	124	135	128	129	124	121	118	123	126	128
134	128	128	130	125	125	127	122	122	127	127	127	128
134	129	124	126	123	121	126	127	127	127	124	129	128
128	124	124	127	121	127	127	125	129	128	129	127	123
120	127	125	130	131	129	127	121	123	123	127	126	128
128	128	125	123	121	127	125	128	134	128	129	125	122
121	120	124	127	125	130	129	128	130	124	127	123	120
125	127	124	128	128	128	130	126	124	124	125	126	126
126	127	126	128	131	127	127	120	125	124	126	127	127
127	128	131	125	124	127	121	126	127	124	128	130	126
126	126	127	126	126	125	126	126	127	127	124	131	127
124	125	125	126	126	127	127	126	125	127	127	126	126
125	126	126	127	127	127	126	125	125	125	126	126	127
127	127	127	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	126	126	126	126	126	126	126	126

126	126	126	126	126	126	126	126	126	126	126	126	126
126	126	126	126	126	128	129	130	131	132	133	134	135
136	137	138	139	140	141	142	143	144	145	146	147	148
149	150	151	152	153	154	155	156	157	158	159	160	161
162	163	164	165	166	167	168	169	170	171	172	173	174
175	176	177	178	179	180	181	182	183	184	185	186	187
188	189	190	191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210	211	212	213
214	215	216	217	218	219	220	221	222	223	224	225	226
227	228	229	230	231	232	233	234	235	237	237	239	239
241	241	243	243	245	245	247	247	249	249	251	251	253
253	255	255										

Montages à composants programmables

APPLICATIONS PRATIQUES

Mémoires EPROM ou EEPROM, réseaux logiques programmables PAL, et même cartes à puce sont désormais des composants courants et peu coûteux.

Grâce à des équipements de programmation pouvant être fort simples, il est facile de les transformer en véritables *circuits intégrés spécifiques*, spécialement conçus pour une application précise.

Il devient ainsi possible de construire très simplement des montages performants qui seraient difficilement réalisables à partir de composants standards : contrôleurs universels pour automatismes, synthétiseurs de sons personnalisés, serrures électroniques à cartes, systèmes de sécurité et même, pourquoi pas, gadgets divertissants.

Ce livre contient tout ce qu'il faut pour réaliser tout un choix de tels montages : schémas, nomenclatures, plans de câblage sur circuits imprimés, et toutes les données pour personnaliser les composants programmables.

Patrick GUEULLE

EDITIONS TECHNIQUES ET SCIENTIFIQUES FRANÇAISES

ETSF

9 782855 352107

Code 23901

ISBN 2-85535-210-X